Video analysis-based vehicle detection and tracking using an MCMC sampling framework

Jon Arróspide*[1], Luis Salgado[1] and Marcos Nieto[2]

[1]Escuela Técnica Superior de Ingenieros de Telecomunicación, Universidad Politécnica de Madrid, Grupo de Tratamiento de Imágenes, Madrid 28040, Spain

[2]Vicomtech-IK4, Research Alliance, San Sebastián 20009, Spain

*Corresponding author: jal@gti.ssr.upm.es

Email addresses:

LS: lsa@gti.ssr.upm.es

MN: mnieto@vicomtech.org

Abstract This article presents a probabilistic method for vehicle detection and tracking through the analysis of monocular images obtained from a vehicle-mounted camera. The method is designed to address the main shortcomings of traditional particle filtering approaches, namely Bayesian methods based on importance sampling, for use in traffic environments. These methods do not scale well when the dimensionality of the feature space grows, which creates significant limitations when tracking multiple objects. Alternatively, the proposed method is based on a Markov chain

Monte Carlo (MCMC) approach, which allows efficient sampling of the feature space. The method involves important contributions in both the motion and the observation models of the tracker. Indeed, as opposed to particle filter-based tracking methods in the literature, which typically resort to observation models based on appearance or template matching, in this study a likelihood model that combines appearance analysis with information from motion parallax is introduced. Regarding the motion model, a new interaction treatment is defined based on Markov random fields (MRF) that allows for the handling of possible inter-dependencies in vehicle trajectories. As for vehicle detection, the method relies on a supervised classification stage using support vector machines (SVM). The contribution in this field is twofold. First, a new descriptor based on the analysis of gradient orientations in concentric rectangles is defined. This descriptor involves a much smaller feature space compared to traditional descriptors, which are too costly for real-time applications. Second, a new vehicle image database is generated to train the SVM and made public. The proposed vehicle detection and tracking method is proven to outperform existing methods and to successfully handle challenging situations in the test sequences.

Keywords: Object tracking; Monte Carlo methods; intelligent vehicles; HOG.

1 Introduction

Signal processing techniques have been widely used in sensing applications to automatically characterize the environment and understand the scene. Typical problems include ego-motion estimation, obstacle detection, and object localization, monitoring, and tracking, which are usually addressed by processing the information coming from sensors such as radar, LIDAR, GPS, or video-cameras. Specifically, methods based on video analysis play an important role due to their low cost, the striking increase of processing capabilities, and the significant advances in the field of computer vision.

Naturally object localization and monitoring are crucial to have a good understanding of the scene. However, they have an especially critical role in safety applications, where the objects may constitute a threat to the observer or to any other individual. In particular, the tracking of vehicles in traffic scenarios from an on-board camera constitutes a major focus of scientific and commercial interest, as vehicles cause the majority of accidents.

Video-based vehicle detection and tracking have been addressed in a variety of ways in the literature. The former aims at localizing vehicles by exhaustive search in the images, whereas the latter aims to keep track of already detected vehicles. As regards vehicle detection, since exhaustive image search is costly, most of the methods in the literature proceed in a two-stage fashion: hypothesis generation, and hypothesis verification. The first usually involves a rapid search, so that the image regions that do not match an expected feature of the vehicle are disregarded, and only a small

number of regions potentially containing vehicles are further analyzed. Typical features include edges [1], color [2,3], and shadows [4]. Many techniques based on stereovision have also been proposed (e.g., [5,6]), although they involve a number of drawbacks compared to monocular methods, especially in terms of cost and flexibility.

Verification of hypotheses is usually addressed through model-based or appearance-based techniques. The former exploit *a priori* knowledge of the structure of the vehicles to generate a description (i.e., the model) that can be matched with the hypotheses to decide whether they are vehicles or not. Both rigid (e.g., [7]) and deformable (e.g., [8]) vehicle models have been proposed. Appearance-based techniques, in contrast, involve a training stage in which features are extracted from a set of positive and negative samples to design a classifier. Neural networks [9] and support vector machines (SVM) [10,11] are extensively used for classification, while many different techniques have been proposed for feature extraction. Among others, histograms of oriented gradients (HOG) [12,13], principal component analysis [14], Gabor filters [11] and Haar-like features [15,16] have been applied to derive the feature set for classification.

Direct use of many of these techniques is very time-consuming and thus unrealistic in real-time applications. Therefore, in this study we propose a vehicle detection method that exploits the intrinsic structure of the vehicles in order to achieve good detection results while involving a small feature space (and hence low computational overhead). The method combines prior

knowledge on the structure of the vehicle, based on the analysis of vertical symmetry of the rear, with appearance-based feature training using a new HOG-based descriptor and SVM. Additionally, a new database containing vehicle and non-vehicle images has been generated and made public, which is used to train the classifier. The database distinguishes between vehicle instances depending on their relative position with respect to the camera, and hence allows for an adaptation of the feature selection and the classifier in the training phase according to the vehicle pose.

In regard to object tracking, feature-based and model-based approaches have been traditionally utilized. The former aim to characterize objects by a set of features (e.g., corners [17] and edges [18] have been used to represent vehicles) and to subsequently track them through inter-frame feature matching. In contrast, model-based tracking uses a template that represents a typical instance of the object, which is often dynamically updated [19, 20]. Unfortunately, both approaches are prone to errors in traffic environments due to the difficulty in extracting reliable features or in providing a canonical pattern of the vehicle.

To deal with these problems, many recent approaches to object tracking entail a probabilistic framework. In particular, the Bayesian approach [21, 22], especially in the form of particle filtering, has been used in many recent studies (e.g., [23–25]), to model the inherent degree of uncertainty in the information obtained from image analysis. Bayesian tracking of multiple objects can be found in the literature both using individual Kalman or

particle filters (PF) for each object [24,26] and a joint filter for all of the objects [27,28]. The latter is better suited for applications in which there is some degree of interaction among objects, as it allows for the controlling of the relations among objects in a common dynamic model (those are much more complicated to handle through individual PF [29]). Notwithstanding, the computational complexity of joint-state traditional importance sampling strategies grows exponentially with the number of objects, which results in a degraded performance with respect to independent PF-based tracking when there are several participants (as occurs in a traffic scenario). Some recent studies, especially relating to radar/sonar tracking applications [30], resort to finite set statistics (FISST) and use random sets rather than vectors to model multiple objects state, which is especially suitable for the cases where the number of objects is unknown.

On the other hand, PF-based object tracking methods found in the literature resort to appearance information for the definition of the observation model. For instance, in [23], a likelihood model comprising edge and silhouette observation is employed to track the motion of humans. In turn, the appearance-based model used in [27] for ant tracking consists of simple intensity templates. However, methods using appearance-only models are only bound to be successful under controlled scenarios, such as those in which the background is static. In contrast, the considered on-board traffic monitoring scenarios entail a dynamically changing background and varying illumination conditions, which affect the appearance of the vehicles.

In this study, we present a new framework for vehicle tracking which combines efficient sampling, handling of vehicle interaction, and reliable observation modeling. The proposed method is based on the use of Markov chain Monte Carlo (MCMC) approach to sampling (instead of the traditional importance sampling) which renders joint state modeling of the objects affordable, while also allowing to easily accommodate interaction modeling. In effect, driver decisions are affected by neighboring vehicle trajectories (vehicles tend to occupy free space), and thus an interaction model based on Markov random fields (MRF) [31] is introduced to manage inter-vehicle relations. In addition, an enriched observation model is proposed, which fuses appearance information with motion information. Indeed, motion is an inherent feature of vehicles and is considered here through the geometric analysis of the scene. Specifically, the projective transformation relating the road plane between consecutive time points is instantaneously derived and filtered temporally based on a data estimation framework using a Kalman filter. The difference between the current image and the previous image warped with this projectivity allows for the detection of regions likely featuring motion. Most importantly, the combination of appearance and motion-based information provides robust tracking even if one of the sources is temporarily unreliable or unavailable. The proposed system has been proven to successfully track vehicles in a wide variety of challenging driving situations and to outperform existing methods.

2 Problem statement and proposed framework

As explained in Section 1, the proposed tracking method is grounded on a Bayesian inference framework. Object tracking is addressed as a recursive state estimation problem in which the state consists of the positions of the objects. The Bayesian approach allows for the recursive updating of the state of the system upon receipt of new measurements. If we denote $\mathbf{s}_k$ the state of the system at time k and $\mathbf{z}_k$ the measurement at the same instant, then Bayesian theory provides an optimal solution for the posterior distribution of the state given by

$$p(\mathbf{s}_k|\mathbf{z}_{1:k}) = \frac{p(\mathbf{z}_k|\mathbf{s}_k)\int p(\mathbf{s}_k|\mathbf{s}_{k-1})p(\mathbf{s}_{k-1}|\mathbf{z}_{1:k-1})d\mathbf{s}_{k-1}}{p(\mathbf{z}_k|\mathbf{z}_{1:k-1})} \tag{1}$$

where $\mathbf{z}_{1:k}$ integrates all the measurements up to time k [21]. Unfortunately, the analytical solution is intractable except for a set of restrictive cases. Particularly, when the state sequence evolution is a known linear process with Gaussian noise and the measurement is a known linear function of the state (also with Gaussian noise) then the Kalman filter constitutes the optimal algorithm to solve the Bayesian tracking problem. However, these conditions are highly restrictive and do not hold for many practical applications. Hence, a number of suboptimal algorithms have been developed to approximate the analytical solution. Among them, particles filters (also known as bootstrap filtering or condensation algorithm) play an outstanding role and have been used extensively to solve problems of a very different nature. The key idea of particles filters is to represent the posterior probability density

function by a set of random discrete samples (called particles). In the most common approach to particle filtering, known as importance sampling, the samples are drawn independently from a proposal distribution $q(\cdot)$, called importance density.

However, importance sampling is not the only approach to particle filtering. In particular, MCMC methods provide an alternative framework in which the particles are generated sequentially in a Markov chain. In this case, all the samples are equally weighed and the solution in (1) can therefore be approximated as

$$p(\mathbf{s}_k|\mathbf{z}_{1:k}) \approx c \cdot p(\mathbf{z}_k|\mathbf{s}_k) \sum_{r=1}^{N} p(\mathbf{s}_k|\mathbf{s}_{k-1}^{(r)}) \tag{2}$$

where the state of the rth particle at time k is denoted $\mathbf{s}_k^{(r)}$, N is the number of particles, and c is the inverse of the evidence factor in the denominator of (1). As opposed to importance sampling, a record of the current state is kept, and each new sample is generated from a proposal distribution that depends on the current sample, thus forming a Markov chain. The proposal distribution is usually chosen to be simple so that samples can easily be drawn. The advantage of MCMC methods is that the complexity increases only linearly with the number of objects, in contrast to importance sampling, in which the complexity grows exponentially [27]. This implies that using the same computational resources, MCMC will be able to generate a larger number of particles and hence, better approximate the posterior distribution. The potential of MCMC has been shown for processing data of different sensors, e.g., for target tracking in radar [32] or video-based ant

tracking [27]. An MCMC framework is thus used in this study for vehicle tracking.

This framework requires that the observation model, $p(\mathbf{z}_k|\mathbf{s}_k)$, and the dynamic or motion model, $p(\mathbf{s}_k|\mathbf{s}_{k-1})$, be defined. Selection of these models is a key aspect to the performance of the framework. In particular, in order to define a scheme that can be lead to improved performance in an MCMC-based Bayesian framework, we have tried to first identify the weaknesses of the state-of-the-art methods related to the definition of these models. Regarding the observation model, as stated in Section 1, most methods in the literature resort to appearance-based models typically using templates or some features that characterize the objects of interest. Although this kind of models perform well when applied to controlled scenarios, they prove insufficient for the traffic scenario. In this environment the background changes dynamically, and so do weather and illumination conditions, which limits the effectiveness of appearance-only models. In addition, the appearance of vehicles themselves is very heterogenous (e.g., color, size), thus making their modeling much more challenging.

These limitations in the design of the observation model are addressed twofold. First, rather than the usual template matching methods, a probabilistic approach is taken to define the appearance-based observation model using the Expectation-Maximization technique for likelihood function optimization. Additionally, we extend the observation model so that it not only includes a set of appearance-based features, but also considers a feature

that is inherent to vehicles, i.e., their motion, so that it is more robust to changes in the appearance of the objects. In particular, the model for the observation of motion is based on the temporal alignment of the images in the sequence through the analysis of multiple-view geometry.

As regards the motion model, it is designed under the assumption that vehicles velocity can be approximated to be locally constant, which is valid in highway environments. As a result, the evolution of a vehicle's position can be traced by a first-order linear model. However, linearity is lost due to the perspective effect in the acquired image sequence. To preserve linearity we resort to a plane rectification technique, usually known as inverse perspective mapping (IPM) [33]. This computes the projective transformation, T, that produces an aerial or bird's-eye view of the scene from the original image. The image resulting from plane rectification will be referred to as the rectified domain or the transformed domain. In the rectified domain, the motion of vehicles can be safely described as a first-order linear equation with an added random noise.

One important limitation regarding the dynamic model in existing methods is the interaction treatment. Most approaches to multiple vehicle tracking involve independent motion models for each vehicle. However, this requires an external method for handling of interaction, and often this is simply disregarded. In contrast, we have designed an MRF-based interaction model that can be easily integrated with the above-mentioned individual vehicle dynamic model.

Finally, a method is necessary to detect new vehicles in the scene, so that they can be integrated in the tracking framework. This is addressed in the current work by using a two-step procedure composed of an initial hypothesis generation and a subsequent hypothesis verification. In particular, candidates are verified using a supervised classification strategy over a new descriptor based on HOG features. The proposed feature descriptor and the classification strategy are explained in Section 6.

The explained framework is summarized in the general scheme shown in Fig. 1. The scheme shows the main constituent blocks of the method, i.e., observation model (which in turn relies on appearance and motion analysis), motion model, vehicle tracking algorithm, and new vehicle detection algorithm, as well as the techniques used for their design. These blocks are explained in detail in the following sections.

3 Vehicle tracking algorithm

The designed vehicle tracking algorithm aims at estimating the position of the vehicles existing at each time of the image sequence. Hence, the state vector is defined to comprise the position of all the vehicles $\mathbf{s}_k = \{s_{i,k}\}_{i=1}^{M}$, where $s_{i,k}$ denotes the position of vehicle i, and M is the number of vehicles existing in the image at time k. As stated, the position of a vehicle is defined in the rectified domain given by the transformation T, although back-projection to the original domain is naturally possible via the inverse projective transformation T^{-1}.

An example of the bird's-eye view obtained through IPM is illustrated in Fig. 2. Observe that the upper part of the vehicles is distorted in the rectified domain. This is due to the fact, that IPM calculates the appropriate transformation for a given reference plane (in this case the road plane), which is not valid for all of the elements outside this plane. Therefore, analysis is focused on the road plane image and the position of a vehicle will be defined as the middle point of its lower edge. This is given in pixels, $s_{i,k} = (x_{i,k}, y_{i,k})$, where x indicates the column and y is the row of the corresponding point in the image, while the origin is set at the upper-left corner of the image.

In order to estimate the joint state of all of the vehicles, the MCMC method is applied. As mentioned, in MCMC the approximation to the posterior distribution of the state is given by (2), which, assuming that the likelihood of the different objects is independent, can be rewritten as follows:

$$p(\mathbf{s}_k|\mathbf{z}_{1:k}) \approx c \cdot \prod_{i=1}^{M} p(z_{i,k}|s_{i,k}) \sum_{r=1}^{N} p(\mathbf{s}_k|\mathbf{s}_{k-1}^{(r)}) \tag{3}$$

where $z_{i,k}$ is the observation at time k for object i. In MCMC, samples are generated sequentially from a proposal distribution that depends on the currents state, therefore the sequence of samples forms a Markov chain. The Markov chain of samples at time k is generated as follows. First, the initial state is obtained as the mean of the samples in $k-1$, $\mathbf{s}_k^0 = \sum_r \mathbf{s}_{k-1}^{(r)}/N$. New samples for the chain are generated from a proposal distribution $Q(\cdot)$. Specifically, we follow a Gibbs-like approach, in which only one target is

changed at each step of the chain. At step τ the proposed position $s'_{i,k}$ of the randomly selected target i is thus sampled from the proposal distribution, which in our case is a Gaussian centered at the value of the last sample for that target, $Q(s'_{i,k}|s^{(\tau)}_{i,k}) = \mathcal{N}(s'_{i,k}|s^{(\tau)}_{i,k}, \sigma_q)$. The candidate sample is therefore $\mathbf{s}'_k = (\mathbf{s}^{(\tau)}_{\setminus i,k}, s'_{i,k})$, where $\mathbf{s}_{\setminus i,k}$ denotes $\mathbf{s}_k$ but with $s_{i,k}$ omitted. This sample is or is not accepted according to the Metropolis algorithm, which evaluates the posterior probability of the candidate sample in comparison to that of the previous sample and defines the following probability of acceptance [31]:

$$A(\mathbf{s}'_k, \mathbf{s}^{(\tau)}_k) = \min\left(1, \frac{p(\mathbf{s}'_k|\mathbf{z}_{1:k})}{p(\mathbf{s}^{(\tau)}_k|\mathbf{z}_{1:k})}\right) \tag{4}$$

This implies that, if the posterior probability of the candidate sample is larger than that of $\mathbf{s}^{(\tau)}_k$ the candidate sample is accepted, and if it is smaller, it is accepted with probability equal to the ratio between them. The latter case can be readily simulated by selecting a random number t from a uniform distribution over the interval $(0, 1)$, and then accepting the candidate sample if $A(\mathbf{s}'_k, \mathbf{s}^{(\tau)}_k) > t$. In the case of acceptance, $\mathbf{s}^{(\tau+1)}_k = \mathbf{s}'_k$. Otherwise the previous sample is repeated $\mathbf{s}^{(\tau+1)}_k = \mathbf{s}^{(\tau)}_k$.

Observe that the samples obtained with the explained procedure are highly correlated. It is a common practice to retain only every Lth sample and leave out the rest, which is called thin-out. In addition, the first B samples are discarded to prevent the estimation from being degraded by bad initialization. Finally, at each time step the vehicle position estimates, $\bar{\mathbf{s}}_k = \{\bar{s}_{i,k}\}_{i=1}^{M}$, are inferred as the mean of the valid particles $\mathbf{s}^{(r)}_k$:

$$\bar{\mathbf{s}}_k = \frac{1}{N}\sum_{r=1}^{N}\mathbf{s}_k^{(r)} \tag{5}$$

3.1 Summary of the sampling algorithm

The previously introduced sampling process can be summarized as follows. At time k we want to obtain a set of samples, $\{\mathbf{s}_k^{(r)}\}_{r=1}^N$, which approximate the posterior distribution of the vehicles state. In order to obtain those, we make use of the samples at the previous time step, $\{\mathbf{s}_{k-1}^{(r)}\}_{r=1}^N$, and of the motion and likelihood models, within the MCMC sampling framework. The steps of the sampling algorithm at time k are the following:

(1) The average of the particles at the previous time step is taken as the initial state of the Markov chain: $\mathbf{s}_k^0 = \sum_r \mathbf{s}_{k-1}^{(r)}/N$.

(2) To generate each new sample of the chain, $\mathbf{s}_k^{(\tau+1)}$, an object i is picked randomly and a new state $s'_{i,k}$ is proposed for it by sampling from the proposal distribution, $Q(s'_{i,k}|s_{i,k}^{(\tau)}) = \mathcal{N}(s'_{i,k}|s_{i,k}^{(\tau)}, \sigma_q)$. Since the other targets remain unchanged, the candidate joint state is $\mathbf{s}'_k = (\mathbf{s}_{\setminus i,k}^{(\tau)}, s'_{i,k})$.

(3) The posterior probability estimate of the proposed sample, $p(\mathbf{s}'_k|\mathbf{z}_{1:k})$, is computed according to the Equation (3), which depends on both the motion and the observation models. The motion model, $p(\mathbf{s}_k|\mathbf{s}_{k-1})$, is given by Equation (9), while the observation model for a vehicle, $p(z_{i,k}|s_{i,k})$, is specified in (22).

(4) The candidate sample $\mathbf{s}'_k$ is accepted with probability $A(\mathbf{s}'_k, \mathbf{s}_k^{(\tau)})$ computed as in Equation (4). In the case of acceptance, the new sample of

the Markov chain is $\mathbf{s}_k^{(\tau+1)} = \mathbf{s}'_k$, otherwise the previous sample is copied, $\mathbf{s}_k^{(\tau+1)} = \mathbf{s}_k^{(\tau)}$.

(5) Finally, only one every Lth samples is retained to avoid excessive correlation, and the first B samples are discarded. The final set of N samples provides an estimate of the posterior distribution and the vehicle position estimates are computed as the average of the samples as in Equation (5).

4 Motion and interaction model

The motion model is defined in two steps: the first layer deals with the individual movement of a vehicle in the absence of other participants, and the second layer addresses the movement of vehicles in a common space. The tracking condition involves the assumption that vehicles are moving on a planar surface (i.e., the road) with a locally constant velocity. This is a very common assumption at least in highway environments, and allows to formulate tracking of vehicle positions with a first-order linear model. Although linearity is lost in the original image sequence, due to the position of the camera, which creates a given perspective of the scene, as stated in Section 2 it can be retrieved by using IPM and working in the rectified domain. Hence, the evolution of a vehicle position in time, $s_{i,k} = (x_{i,k}, y_{i,k})$, is modeled with a first-order linear equation in both coordinates:

$$s_{i,k} = s_{i,k-1} + \tilde{v}_{i,k} \Delta t + m_k \tag{6}$$

where Δt is the elapsed time between frames, $\tilde{v}_{i,k}$ is the prediction of the vehicle velocity at time k derived from the previous positions as $\tilde{v}_{i,k} =$

$(s_{i,k-1} - s_{i,k-L})/(L \cdot \Delta t)$, and $m_k = (m_k^x, m_k^y)$ comprises i.i.d. Gaussian distributions corresponding to noise in the x and y coordinates of the motion model:

$$p(m_k^x) \sim \mathcal{N}(0, \sigma_m^x)$$

$$p(m_k^y) \sim \mathcal{N}(0, \sigma_m^y)$$

In particular, from the experiments performed on the test sequences, noise variances are heuristically set to $\sigma_m^x = 10$ and $\sigma_m^y = 15$. The individual dynamic model can thus be reformulated as

$$p(s_{i,k}|s_{i,k-1}) = \mathcal{N}(s_{i,k}|s_{i,k-1} + \tilde{v}_{i,k}\Delta t, \sigma_m) \tag{7}$$

where $\sigma_m = (\sigma_m^x, \sigma_m^y)$.

Once the expected evolution of each individual target has been defined, their interaction must also be accounted for in the model. A common approach to address interaction is through MRFs, which graphically represent a set of conditional independent relations. An MRF (also known as undirected graph) is composed of a set of nodes V, which represent the variables, and a set of links representing the relations among them. The joint distribution of the variables can be factorized as a product of functions defined over subsets of connected nodes (called cliques, $\mathbf{x}_C$). These functions are known as potential functions and denoted $\phi_C(\mathbf{x}_C)$. In the proposed MRF, the nodes V_i (representing the vehicle positions $s_{i,k} = (x_{i,k}, y_{i,k})$) are connected according to a distance-based criterion. Specifically, if two vehicles, i and j, are at a distance smaller than a predefined threshold, then the nodes

representing the vehicles are connected and form a clique. The potential function of the clique is defined as

$$\phi_C(\mathbf{x}_C) = 1 - \exp\left(-\frac{\alpha_x \delta x^2}{w_l^2}\right) \exp\left(-\frac{\alpha_y \delta y^2}{d_s^2}\right) \tag{8}$$

where $\delta x = |x_{i,k} - x_{j,k}|$ and $\delta y = |y_{i,k} - y_{j,k}|$. The functions $\phi_C(\mathbf{x}_C)$ can be regarded as penalization factors that decrease the joint probability of a hypothesized state if it involves an unexpected relations among targets. Potential functions consider the expected width of the lane, w_l, and the longitudinal safety distance, d_s. In addition, the design parameters α_x and α_y are selected so that $\alpha_x = 0.5$ and $\alpha_y = 0.5$ whenever a vehicle is at a distance $\delta x = w_l/4$ or $\delta y = d_s$ of another vehicle. Finally, the joint probability is given by the product of the individual probabilities associated to each node and the product of potential functions in existing cliques:

$$p(\mathbf{s}_k|\mathbf{s}_{k-1}) = \prod_{i=1}^{M} p(s_{i,k}|s_{i,k-1}) \prod_{\mathcal{C}} \phi_C(\mathbf{x}_C) \tag{9}$$

where $\mathcal{C}$ is the set of the two-node cliques. Let us now introduce this motion model in the expression of the posterior distribution in (2):

$$p(\mathbf{s}_k|\mathbf{z}_{1:k}) \approx c \cdot p(\mathbf{z}_k|\mathbf{s}_k) \sum_{r=1}^{N} \prod_{i=1}^{M} p(s_{i,k}|s_{i,k-1}^{(r)}) \prod_{C} \phi_C(\mathbf{x}_C) \tag{10}$$

It is important to note that the potential factor does not depend on the previous state, therefore (10) can be rewritten as

$$p(\mathbf{s}_k|\mathbf{z}_{1:k}) \approx c \cdot p(\mathbf{z}_k|\mathbf{s}_k) \prod_{C} \phi_C(\mathbf{x}_C) \sum_{r=1}^{N} \prod_{i=1}^{M} p(s_{i,k}|s_{i,k-1}^{(r)}) \tag{11}$$

Modeling of vehicle interaction thus requires only the evaluation of an additional factor in the posterior distribution, while producing significant gain in the tracking performance.

5 Observation model

5.1 Appearance-based analysis

The first part of the observation model deals with the appearance of the objects. The aim is to obtain the probability $p_a(z_{i,k}|s_{i,k})$ of the current appearance observation given the object state $s_{i,k}$ (note the subscript a that denotes "appearance"). In other words, we would like to know if the current appearance-related measurements support the hypothesized object state. In order to derive the probability $p_a(z_{i,k}|s_{i,k})$ we will proceed in two levels. First, the probability that a pixel belongs to a vehicle will be defined according to the observation for that pixel. Second, by analyzing the pixel-wise information around the position given by $s_{i,k}$, the final observation model will be defined at region level.

The pixel-wise model aims to provide the probability that a pixel belongs to a vehicle. This will be addressed as a classification problem, and it is therefore necessary to define the different categories expected in the image. In particular, the rectified image (see example in Fig. 2.) contains mainly three types of elements: vehicles, road pavement, and lane markings. A fourth class will also be included in the model to account for any other kind of elements (such as median stripes or guard rails).

The Bayesian approach is adopted to address this classification problem. Specifically, the four classes are denoted by $\mathcal{S} = \{P, L, V, U\}$, which corresponds to the pavement, lane markings, vehicles, and unidentified elements. Let us also denote X_i the event that a pixel x is classified as belonging to

the class $i \in \mathcal{S}$. Then, if the current measurement for pixel x is represented by z_x, the posterior probability that the pixel x corresponds to X_i is given by the Bayes rule

$$P(X_i|z_x) = \frac{p(z_x|X_i)P(X_i)}{P(z_x)} \tag{12}$$

where $p(z_x|X_i)$ is the likelihood function, $P(X_i)$ is the prior probability of class X_i, and $P(z_x)$ is the evidence, computed as $P(z_x) = \sum_{i \in \mathcal{S}} p(z_x|X_i)P(X_i)$, which is a scale factor that ensures that the posterior probabilities sum to one. Likelihoods and prior probabilities are defined in the following section.

5.1.1 Likelihood functions

In order to construct the likelihood functions, a set of features have to be defined that constitute the current observation regarding appearance. These features should achieve a high degree of separation between classes while, at the same time, be significant for a broad set of scenarios. In general terms, the following considerations hold when analyzing the appearance of the bird's-eye view images. First, the road pavement is usually homogeneous with slight intensity variations among pixels. In turn, lane markings constitute near-vertical stripes of high-intensity, surrounded by regions of lower intensity. As for vehicles, they typically feature very low intensity regions in their lower part, due to vehicle's shadow and wheels. Hence, two features are used for the definition of the appearance-based likelihood model, namely the intensity value, I_x, and the response to a lane-marking detector, R_x. For the latter, any of the methods available in the literature can be utilized [33, 34]. For this work, a lane marking detector similar to that presented in [35]

is used, whose response is defined in every row of the image as

$$R_x = 2I_x - (I_{x-\tau} + I_{x+\tau}) \tag{13}$$

where τ is the expected width of a lane marking in the rectified domain. The likelihood models are defined as parametric functions of these two features. In particular, they are modeled as Gaussian probability density functions:

$$p(I_x|X_i) = \frac{1}{\sqrt{2\pi}\sigma_{I,i}} \exp\left(-\frac{1}{2\sigma_{I,i}^2}(I_x - \mu_{I,i})^2\right) \tag{14}$$

$$p(R_x|X_i) = \frac{1}{\sqrt{2\pi}\sigma_{R,i}} \exp\left(-\frac{1}{2\sigma_{R,i}^2}(R_x - \mu_{R,i})^2\right) \tag{15}$$

where the parameters for the intensity and the lane marking detector are denoted respectively by the subscripts 'I' and 'R'. Specifically, the distribution of the class corresponding to unidentified elements, which would intuitively be uniformly distributed for both features, is instead also modeled as a Gaussian of very high fixed variance to ease further processing. Additionally, likelihood functions are assumed to be conditionally independent on these features for all the classes X_i, thus it is

$$p(z_x|X_i) = p(I_x|X_i)p(R_x|X_i) \tag{16}$$

The parameters of the likelihood models in (14) and (15) are estimated via EM. This method is extensively used for solving Gaussian mixture-density parameter estimation (see [36] for details) and is thus perfectly suited to the posed problem. In particular, it provides an analytical maximum likelihood solution that is found iteratively. In addition, it is simple,

easy to implement and converges quickly to the solution when a good initialization is available. In this case, this is readily available from the previous frame, that is, the results from the previous image can recursively be used as starting point in each incoming image. The data distribution is given by

$$p(I_x) = \sum_{i \in \mathcal{S}} p(X_i) p(I_x | X_i) \tag{17}$$

$$p(R_x) = \sum_{i \in \mathcal{S}} p(X_i) p(R_x | X_i) \tag{18}$$

Since the densities of the features I_x and R_x are independent, the optimization is carried out separately for these features. Let us first rewrite the expression (17), so that the dependence on the parameters is explicit:

$$p(I_x | \Theta_I) = \sum_{i \in \mathcal{S}} \omega_{I,i} p(I_x | \Theta_{I,i}) \tag{19}$$

where $\Theta_{I,i} = \{\mu_{I,i}, \sigma_{I,i}\}$ and $\Theta_I = \{\Theta_{I,i}\}_{i \in P,L,V}$. Observe that the prior probabilities have been substituted by factors $\omega_{I,i}$ to adopt the notation typical of mixture models. The set of unknown parameters is composed of the parameters of the densities and of the mixing coefficients, $\Theta = \{\Theta_{I,i}, \omega_{I,i}\}_{i \in P,L,V}$. Thereby, the parameters resulting from the final EM iteration are fed into the Bayesian model defined in Equations (12)–(15). The process is completely analogous for the feature R_x.

5.1.2 Appearance-based likelihood model

The result of the proposed appearance-based likelihood model is a set of pixel-wise probabilities of each of the classes. Naturally, in order to know the likelihood of the current object state candidate, we must evaluate the

region around the vehicle position given by $s_{i,k} = (x_{i,k}, y_{i,k})$. The vehicle position has been defined as the midpoint of its lower edge (i.e., the segment delimiting the transition from road to vehicle). Hence, we expect that in the neighborhood above $s_{i,k}$, pixels display high probability to belong to the vehicle class, $p(X_V|z_x)$, while the neighborhood below $s_{i,k}$ should involve low vehicle probabilities if the candidate state is good. Therefore, the appearance-based likelihood of the object state $s_{i,k}$ is defined as

$$p_a(z_{i,k}|s_{i,k}) = \frac{1}{(w+1)h}\left(\sum_{x\in R_a} p(X_V|z_x) + \sum_{x\in R_b}(1-p(X_V|z_x))\right)$$

where R_a is the region of size $(w+1)\times h/2$ above $s_{i,k}$, $R_a = \{x_{i,k} - w/2 \leq x < x_{i,k} + w/2; y_{i,k} - h/2 \leq y < y_{i,k}\}$, and R_b is the region of the same size below $s_{i,k}$, $R_b = \{x_{i,k} - w/2 \leq x < x_{i,k} + w/2; y_{i,k} < y \leq y_{i,k} + h/2\}$.

5.2 Motion-based analysis

As mentioned above, the second source of information for the definition of the likelihood model is motion analysis. Two-view geometry fundamentals are used to relate the previous and current views of the scene. In particular, the homography (i.e., projective transformation) of the road plane is estimated between these two points in time. This allows us to generate a prediction of the road plane appearance in future instants. However, vehicles (which are generally the only objects moving on the road plane) feature inherent motion in time, hence their projected position in the plane differs from that observed. The regions involving motion are identified through im-

age alignment of the current image and the previous image warped with the homography. These regions will correspond to vehicles with high probability.

5.2.1 Homography calculation

The first step toward image alignment is the calculation of the road plane homography between consecutive frames. As shown in [37] the homography that relates the points of a plane between two different views can be obtained from a minimum of four feature correspondences by means of the direct linear transformation (DLT). Indeed, in many applications the texture of the planar object allows to obtain numerous feature correspondences using standard feature extraction and matching techniques, and to subsequently find a good approximation to the underlying homography. However, this is not the case in traffic environments: the road plane is highly homogeneous, and hence most of the points delivered by feature detectors applied on the images belong to background elements or vehicles, and few correspond to the road plane. Therefore, the resulting dominant homography (even if using robust estimation techniques) is in general not that of the road plane.

To overcome this problem, we propose to exploit the specific nature of the environment. In particular, highways are expected to have different kind of markings (mostly lane markings) painted on the road. Therefore, we propose to first use a standard lane marking detector (such as the ones described in [33–35]) and then to restrict the feature search area in extended regions around lane markings. Nevertheless, the resulting set of correspondences will still typically be scarce, and some of them may be incorrect or

inaccurate, depending on the sharpness of the lane marking corners and on the resolution of the image around them. Hence, the instantaneous homography computed from feature correspondences using DLT might be highly unreliable (errors in one of the points will have a large impact in the solution to the equation system of DLT), and sometimes the number of points is not even sufficient to compute it.

For the above-mentioned reasons, intermediate processing of the instantaneous homography is necessary. This is achieved in this study by means of a linear estimation process based on Kalman filtering. Let us first inspect the analytical expression of the homography between two consecutive instants. Figure 3. illustrates the situation of a vehicle with an on-board camera moving on a flat road plane, $\pi_0 = (\mathbf{n}^\top, d)^\top$, where $\mathbf{n} = (0, 1, 0)^\top$ and d is the distance between the camera and the ground plane. The coordinate system of the camera at time k_1 is adopted as the world coordinate system, where Z-axis indicates the driving direction. At time k_2 the camera has moved to position $\mathbf{C}_2$, and rotation $R_x(\alpha)$ might have occurred around the X-axis due to camera shaking (α denotes the change in the pitch angle). Additional rotation $R_y(\beta)$ models variations in the yaw angle (i.e., around the Y-axis), which must be considered in the case the vehicle changes lane or takes a curve. From the previous discussion, and assuming a pinhole camera model, the camera projection matrices at times k_1 and k_2 are, respectively,

$$\begin{aligned} P_1 &= K[I|\mathbf{0}] \\ P_2 &= KR_x(\alpha)R_y(\beta)[I|-\mathbf{C}_2] \end{aligned} \tag{20}$$

The homography H relates the projections, $\mathbf{x}_1$ and $\mathbf{x}_2$, of a 3D point, $\mathbf{X} \in \pi_0$, in two different images. Its expression can be derived from Equation (20). In effect, for the first view it is $\mathbf{x}_1 = P_1\mathbf{X} = K[I|\mathbf{0}]$ and hence any point in the ray $\mathbf{X} = (\mathbf{x}_1^\top (K^{-1})^\top, \rho)^\top$ projects to $\mathbf{x}_1$. The intersection of this ray and the plane π_0 determines the value of the parameter ρ: it is ${\pi_0}^\top \mathbf{X} = \mathbf{n}^\top K^{-1}\mathbf{x} + d\rho = 0$, and thus $\rho = -\mathbf{n}^\top K^{-1}\mathbf{x}_1/d$. The projection $\mathbf{x}_2$ of the point $\mathbf{X}$ into the second view is

$$\begin{aligned}
\mathbf{x}_2 &= P_2\mathbf{X} = KR_x(\alpha)R_y(\beta)[I| - \mathbf{C}_2]\mathbf{X} \\
&= KR_x(\alpha)[R_y(\beta)| - R_y(\beta)\mathbf{C}_2](\mathbf{x}_1^\top (K^{-1})^\top, \rho)^\top \\
&= KR_x(\alpha)[R_y(\beta)K^{-1}\mathbf{x}_1 + \mathbf{t}\rho] \\
&= KR_x(\alpha)[R_y(\beta) - \mathbf{t}\mathbf{n}^\top/d]K^{-1}\mathbf{x}_1
\end{aligned}$$

where $\mathbf{t} = -R_y(\beta)\mathbf{C}_2$. This vector constitutes the translation in the direction in which the vehicle is heading and is thus given by $\mathbf{t} = (0, 0, 1)^\top v/f_r$, where v is the velocity of the vehicle and f_r is the frame rate. From the above equations, the expression of the homography of the plane π_0 between k_1 and k_2 is derived:

$$H = KR_x(\alpha)[R_y(\beta) - \mathbf{t}\mathbf{n}^\top/d]K^{-1} \tag{21}$$

At each time k we have a noisy approximation of the homography H of the road plane between the previous and the current instant. However, the evolution of H in temporal domain is assumed to be smooth due to the intrinsic constraints in vehicle dynamics, therefore better estimates can be obtained by filtering noisy measurements in time. Temporally filtered

estimates of the homography are obtained by modeling H with a zero-order Kalman filter whose state vector is composed of the elements H_{ij} of the homography matrix. The design of the filter is summarized as follows:

$$\mathbf{x}_k^\top = \{H_{ij}, 1 \leq i, j \leq 3\}$$

$$\mathbf{x}_k = \mathbf{x}_{k-1} + \mathbf{w}_k$$

$$\mathbf{z}_k^\top = \{H_{ij}^k, 1 \leq i, j \leq 3\}$$

$$\mathbf{z}_k = \mathbf{x}_k + \mathbf{v}_k$$

The process and measurement noise, $\mathbf{w}_k$ and $\mathbf{v}_k$, are assumed to be given by independent Gaussian distributions, $p(w) \sim N(0, Q)$ and $p(v) \sim N(0, R)$. Observe that the measurement vector is composed of the elements of the instantaneous homography matrix, H^k, computed from image correspondences. As stated above, measurements are expected to be prone to error due to the usually small set of correspondences available, hence the measurement error should be tuned to be larger than the process noise (in the proposed configuration it is $Q = 10^{-6}$, $R = 10^{-3}$).

The designed filter provides corrected estimates for the homography at time k, $\hat{H}^k$, built from the posterior estimate of the filter state, $\hat{\mathbf{x}}_k$. Most importantly, this measure can be used as a prediction for the homography in the next time point. This prediction provides an effective reference to evaluate whether or not the computed instantaneous measurement may be erroneous. Indeed, at the current time k, we can compare the instantaneous homography H^k to the prediction made in the previous time instant $\hat{H}^{k-1}$: if H^k is close to the expected value $\hat{H}^{k-1}$ then the filter equations will be

conveniently updated; in contrast, if the matrices are significantly different, then it is natural to think that noisy correspondences were involved in the calculation of H^k.

The distance between matrices is measured according to the norm of the matrix of differences. Specifically, the norm induced by the 2–norm of a Euclidean space is used. This is obtained by performing singular value decomposition (SVD) of the matrix and retaining its largest singular value [38]. The incoming matrices are accepted and introduced into the Kalman filtering framework only if $\|H^k - \hat{H}^{k-1}\| < t_a$. Otherwise, the measured homography is deemed to be unreliable and the predicted homography is used. The threshold t_a modulates the maximum acceptable distance to the predicted matrix, which depends on the kinematic restrictions of the platform in which the camera is mounted.

In the case of highways, vehicle dynamics are bounded by the maximum speed, the maximum turning angle (i.e., yaw angle, β) and the maximum variation in the pitch angle, α, for a given frame rate. The maximum velocity is considered to be $v = 120$ km/h (33.3 m/s), as enforced by most nation governments. Additionally, a maximum pitch angle variation of $\alpha = \pm 5^\circ$ is considered, and an upper bound of $\beta = \pm 3^\circ$ is inferred for the turning angle according to the standard road geometry design rules. Taking into account these bounds, and assuming an image processing rate of at least 1 fps, the threshold is experimentally found to be $t_a = 60$.

5.2.2 Motion-based likelihood model

Once a time-filtered estimate of the homography $\hat{H}^k$ is available, reliable image alignment can be performed. Image alignment allows for the location of regions of the image likely featuring motion (and therefore likely containing vehicles). The previous image is aligned with the current image by warping it with $\hat{H}^k$. Image alignment is exemplified in Fig. 4. In the upper row, the snapshots of a sequence at times $k-1$ and k are displayed. In Fig. 4.c, the image in Fig. 4.a warped with $\hat{H}^k$ is shown. Observe that this is very similar in the road region to the actual image at time k (Fig. 4.b).

As suggested in the overview of Section 5.2, the reason for image alignment is that all elements in the road plane (except for the points of the vehicle that belong to this plane) are static, and thus their actual position matches that projected by the homography. In contrast, vehicles are moving, hence their positions in the road plane at time k significantly differ from those projected by the homography, which assumes they are static. Therefore, the differences between the image at time k and the image at time $k-1$ warped with $\hat{H}^k$ shall be null for all the elements of the road plane except for the contact zones of the vehicles with the road. The differences in these regions will be more significant when the velocity of the vehicles is high. Fig. 4.d illustrates the difference between the current image—Fig. 4.b—and the previous image warped—Fig. 4.c—for the example referred below. As can be observed, whiter pixels—indicating significant difference—appear in the areas of motion of the vehicles in the road. The transformation of the

elements outside the road is naturally not well represented by $\hat{H}^k$ (this is the homography of the road plane) and thus random regions of strong differences arise in the background, which will be considered clutter.

The pixel-wise difference between the current image and the previous image warped provides information on the likelihood of the current object state candidate, $s_{i,k}$. Analogously to the appearance-based likelihood modeling, the region around the vehicle position indicated by $s_{i,k}$ will be evaluated in order to derive its likelihood. Also, to preserve the duality with the appearance-based analysis, the processing is shifted to the rectified domain using the transformation T defined in Section 2. The resulting image, denoted D_r, is illustrated in Fig. 4.e for the previous example. In particular, the likelihood of belonging to a region of motion is maximum in $x_{\max} = \text{argmax}(D_r(x))$, hence a map of probabilities that the pixel x belongs to a moving region, denoted $p(m|x)$, can be inferred for the whole image as $p(m|x) = D_r(x)/D_r(x_{\max})$.

From the above discussion, observe that the regions of strong differences are between the current vehicle position and the position that it would occupy if it were static (which is always closer to the camera). Therefore, as opposed to the appearance-based modeling (Section 5.1.2), we expect that in the neighborhood below the current vehicle position, $s_{i,k}$, pixels have high likelihood values $p(m|x)$, whereas the neighborhood above x should involve small or null probabilities of motion. Hence, the likelihood of the current

vehicle state $s_{i,k}$ regarding the motion analysis is defined as

$$p_m(z_{i,k}|s_{i,k}) = \frac{1}{(w+1)h}\left(\sum_{x\in R_a}(1-p(m|x)) + \sum_{x\in R_b} p(m|x)\right)$$

where the regions R_a and R_b are those defined in Section 5.1.2, and the subscript m in the probability denotes that it refers to motion observation. The likelihood result obtained from the motion-based analysis is finally combined with that achieved after appearance-based analysis. The joint likelihood of a candidate state $s_{i,k}$ is simply defined as the arithmetic mean of likelihoods:

$$p(z_{i,k}|s_{i,k}) = \frac{1}{2}\left(p_a(z_{i,k}|s_{i,k}) + p_m(z_{i,k}|s_{i,k})\right) \tag{22}$$

Note that, although the product of likelihoods could have been used instead, the mean is preferred in order to avoid that the calculation be biased by likelihood values that are too small.

6 Vehicle detection

Up to this point, the method for vehicle tracking has been explained. However, in normal driving situations it is natural that vehicles come in and out of the field of view of the camera throughout the sequence of images. While management of outgoing vehicles is fairly straightforward (the track simply exceeds the limits of the image), a method for incoming vehicles must be designed. The method proposed in this study follows a two-step approach. In the first stage, hypotheses for vehicle positions are made using the results of appearance-based classification explained in Section 5.1. In

the second, they are verified according to the analysis of a set of features in their associated regions in the original domain.

6.1 Hypothesis generation

Exhaustive search of a certain pattern in the entire image is too time-consuming for applications requiring real-time operation. Hence, it is common to perform some kind of fast pre-processing that restricts the search areas. In this case, we exploit the information extracted from the construction of likelihood models for tracking and use it to generate a set of candidate regions that will be further analyzed. In particular, two types of inputs could be used that correspond to the appearance-based analysis in Section 5.1 and the motion-based analysis in Section 5.2. As referred to in the corresponding section, the latter usually involves noise due to background structures, thus appearance-based information is more suitable for hypothesis generation.

Specifically, based on the appearance analysis, for each pixel the probability that it belongs to a vehicle, $p(X_V|z_x)$ is available. We expect that if there is a new vehicle appearing in the image, a compact zone of high probabilities must be observed in the surrounding of its position. Therefore, in order to localize new vehicles, a binary map B_m is created containing the pixels in which the probability of the vehicle class is larger than that of the other classes, $p(X_V|\mathbf{z}_x) > p(X_i|\mathbf{z}_x)$, $i \in P, L, U$. For example, the binary map obtained for the image in Fig. 5.b is shown in Fig. 5.c. Connected component analysis is performed over B_m to extract the regions with a high

probability of belonging to vehicles. Resulting regions are filtered according to a minimum area criterion in order to remove noise.

Naturally, regions corresponding to the tracked vehicles should exist in B_m. Besides, if there is an additional region in B_m this is regarded as a potential new vehicle in the image and it is further analyzed in the hypothesis verification stage. In particular, in the example in Fig. 5.c three regions are obtained: the upper two regions correspond to existing vehicles, labeled 1 and 2, while the small region in the lower left corner constitutes a potential new vehicle (in this case it is actually a vehicle, as can be observed in Fig. 5.a). Since only the lower part of the vehicles is reliable in the rectified domain, candidates are characterized by their position and width. As potential vehicles are verified according to their appearance in the original image, their position and width in this domain are computed by means of the inverse transformation T^{-1}. Finally, a 1:1 aspect ratio is initially assumed for the vehicle so that a bounding box R_h can be hypothesized for vehicle verification.

6.2 Hypothesis verification

Vehicle verification is based on a supervised classification stage based on SVM. A database of vehicle rear images is generated for the training of the classifier as will be explained in Section 6.2.2. Most importantly the database separates images according to the region in which the vehicle is found (close/middle range in the front, close/middle range in the left,

close/middle range in the right, and far range). Indeed, the view of the vehicle rear changes in these areas and thus affects its intrinsic features. This is taken into account in the design of the feature description, which adapts to the particularities of the different areas. Besides, a different classifier is trained for each of them using the corresponding subsets of images in the database.

As for the feature description, a new descriptor is proposed based on two characteristics that are inherent to the vehicles: high edge content and symmetry. Indeed, the method automatically adapts the area for feature extraction according to a vertical symmetry-based local window refinement. This allows for the correction of position offsets in the hypothesis generation stage and for the adaptation to the vehicle rear contour. Regarding the feature extraction within the refined region, a new descriptor that exploits the inherently rectangular structures of the vehicle rear is designed. The descriptor, called CR-HOG, is based on the analysis of HOG in concentric rectangles around the center of symmetry.

6.2.1 CR-HOG feature extraction:

HOGs evaluate local histograms of image gradient orientations in a dense grid. The underlying idea is that the local appearance and shape of the objects can often be well characterized by the distribution of the local edge directions, even if the corresponding edge positions are not accurately known.

This idea is implemented by dividing the image into small regions called cells. Then, for each cell, a histogram of the gradient orientations over the

pixels is extracted. The original HOG technique, proposed by Dalal and Triggs [12], presents two different kinds of configurations, called Rectangular HOG (R-HOG) and circular HOG (C-HOG), depending on the geometry of the cells used. Specifically, the former involves a grid of rectangular spatial cells and the latter uses cells partitioned in a log-polar fashion.

As stated, in this study we present a new configuration for the cells that better adapts the characteristics of the vehicles. Indeed, the rear of the vehicles presents an inherently rectangular structure: not only is the outer contour of the vehicle rear quasi-rectangular, but the inner structures such as the license plate and the rear window are also rectangular. Hence, we naturally define a new configuration of HOG composed of concentric rectangular cells as shown in Fig. 6.a. This structure will be referred to as CR-HOG (for concentric rectangle-based HOG). The layout of the CR-HOG has five parameters: the number of concentric rectangles b, the number of orientation bins n, the center c_s of the window, its height h_s, and its width w_s.

In practice, the hypothesized region for vehicle verification, R_h, may not perfectly fit the actual bounding box of the vehicle in terms of size and alignment. Specifically, it is often the case that the vehicle is not perfectly centered in R_h, especially on the horizontal axis. Therefore, direct application of CR-HOG (or of standard HOG) over R_h will possibly result in degraded performance. Instead, we refine the region likely containing the vehicle through the analysis of vertical symmetry in the intensity of the

region. In particular, the subregion within R_h giving the maximum degree of vertical symmetry is kept for HOG computing. Vertical symmetry is calculated using the method in [39]. As a result, we obtain the axis of vertical symmetry, x_s and the width of the region that maximizes the symmetry measure, w_s. The height h_s of the window for HOG application is taken as that of R_h and its center is thus given by $c_s = (x_s, h_s/2)$.

Fig. 6.b illustrates the window adaptation approach based on symmetry analysis. Observe that the refined vertical side limits (colored in red) fit much better the bounding edges of the vehicle rears. In practice, the area for calculation of CR-HOG is extended by a 10% so that the outer edges of the vehicle are also accounted for in the descriptor.

The steps for the calculation of CR-HOG on the refined window are the following. First, the gradient magnitude and orientation are computed at each point of the window using a standard operator (Sobel 3×3 masks are used in our implementation). Then, in order to create a histogram of orientations, a number of orientation bins is defined and each pixel votes for the bin that contains its corresponding angle. The votes are weighted by the magnitude of the gradient at that point. Three possible configurations have been considered involving $n = 8, 12$, and 18 bins evenly spaced over [0,180), as illustrated in Fig. 7.

The bins have been shifted so that the vertical and horizontal orientations, which are very frequent in the rear of vehicles, are in the middle of their respective bins. This way, small fluctuations around 0° and 90° will

not affect the descriptor. The histogram of each cell is finally normalized by the area of the cell so that histograms of different cells are in the same order of magnitude. The CR-HOG descriptor is composed of the normalized histogram of orientations of all the cells. The optimal configuration of the number of orientation bins, n, and the number of cells, b, is discussed in Section 7 for each region of the image.

6.2.2 Classification stage:

The CR-HOG descriptors are introduced in a Support Vector Machine-based classifier. A new database containing 4,000 positive vehicle images and 4,000 negative vehicle images is used to train and test the classifier (this can be accessed in the Internet [40]). The core of the database is composed of images from our own collection. Additionally, images have also been extracted from the Caltech [41,42] and the TU Graz-02 [43,44] databases and included in the data set. The joint database consists of images of resolution 64×64 acquired from a vehicle-mounted forward-looking camera. Each image provides a view of the rear of a single vehicle. Some images contain the vehicle completely while others have been drawn to contain it only partially (all images contain at least 50% of the vehicle rear) in order to simulate putative results of the hypothesis generator.

Images involve many different viewpoints of the vehicle rear corresponding to vehicles in different locations relative to the vehicle in which the camera is mounted. Specifically, the space is divided into four main regions: close/middle range in the front, close/middle range in the left, close/middle

range in the right, and far range. The database contains 1,000 images of each of these views. A set of 4,000 images not containing vehicles have also been used to train and test the classifier. The images are selected in such a way that the variability in terms of vehicle appearance, pose, and acquisition conditions (e.g., weather conditions, lighting) is maximized. A classifier based on SVM using linear basis functions is used for each of the four image regions.

7 Experiments and discussion

Experiments regarding the proposed method have been performed twofold. On the one hand, the performance of the novel CR-HOG based approach for vehicle detection is tested on the database referred to in Section 6.2.2. On the other hand, experiments have been carried out in the complete system integrating vehicle detection and tracking over a wide variety of video sequences.

7.1 Experiments on vehicle detection

The SVM-based classifier for vehicle detection explained in Section 6 has been trained and tested in Matlab using the Bioinformatics Toolbox. The method involves two design parameters, namely the number of orientation bins in the histogram, n, and the number of cells, b. Experiments have been performed on the database for values $n = 8, 12, 18$ and $b = 2, 3, 4$. A cross-validation procedure is used to test the method. Specifically, 50% of the

images are randomly selected for the training set and the remaining 50% are used for the testing set. This process is repeated 5 times and the average is computed.

The accuracy or correctly classified rate of samples as a function of these parameters is provided in Table 1 for each of the four regions. These results are graphically represented in Figs. 8. and 9. to facilitate their interpretation. In particular, Fig. 8. shows the accuracy results as a function of the number of cells, b, by averaging on n, for each area of the image. Analogously, Fig. 9. illustrates the average accuracy results as a function of the number of orientation bins, n.

As a first conclusion of the experiments we can infer that the accuracy decreases for $b = 4$ in all the areas of the image. As for the number of orientation bins, a different behavior is observed for the frontal and the sides views. Specifically, for the central close/middle and far ranges similar results are obtained for $n = 8$ and $n = 12$, while the performance decreases notably for $n = 18$. On the contrary, for the left and right areas a significant accuracy increase is observed from $n = 8$ to $n = 12$; a further increase to $n = 18$ does not bring an additional gain. This contrast is indeed reasonable since, from a completely orthogonal viewpoint, the edges of a vehicle are fairly invariant and mostly vertical and horizontal; conversely, in the side views the upright edges corresponding to the back window and its contour (especially the furthest from the image center) tend to divert from vertical-

ity. Consequently, more variability is found in the gradient orientation map, and therefore more bins are necessary to capture fine-detail.

A good trade-off between complexity and performance is achieved by selecting $(b, n) = (2, 8)$ for the close/middle and far ranges, and $(b, n) = (3, 12)$ for the left and right views. This involves respective detection accuracies of 94.88, 85.92, 91.82, and 89.42%, which results in an average correct detection rate of 90.51%. The rate difference between left and right views is due to the particularities of the traffic participants in the right lane, which usually includes slow vehicles (buses, trucks, vans, etc.). These involve a great appearance variety and hence make classification much more challenging. Naturally, the worst classification rate is obtained for the furthest vehicles, in which the edge-details are degraded. The results are improved to an overall accuracy of 92.77% when using a Gaussian radial basis function kernel (instead of the linear kernel), with respective correct detection rates of 96.14, 89.92, 94.14, and 90.86% for the different areas. However, the proposed method continuously generates hypotheses for the potential vehicles. Hence, even if a vehicle is not detected in a given frame, it is usually detected in the following frames. Therefore, the small latency incurred by the linear kernel-based classification is usually negligible and it is not necessary to use more complex kernels.

7.2 Experiments on vehicle tracking

In regard to vehicle tracking, the designed method has been tested on a wide variety of sequences recorded on Madrid, Brussels, and Turin. These sequences, which were acquired in several sessions with different driving conditions (i.e., illumination conditions, weather, pavement color, traffic density, etc.), amount to 22:38 min. Test sequences were acquired from a forward-looking digital video camera installed near the rear mirror of a vehicle driven in highways. The method is able to operate near real-time at 10 fps on average over an Intel(R) Core(TM) i5 processor running at 2.67 GHz. Implementation is done in C++.

The above-mentioned test sequences are used to compare the performance of the proposed tracking method with the two methods most widely used in the literature. Namely, those involve independent tracking of multiple objects with a Kalman filter assigned to each object (which will be referred to as KF-based tracking), or joint tracking using PF based on importance sampling (shortly, SIS-based tracking). For the implementation of KF-based tracking, appearance-based region labeling through connected-component analysis is used as in Section 6.1 to locate vehicles in every frame, and tracks are formed temporally by matching the regions according to a minimum distance criterion. As for SIS-based tracking, a sequential resampling scheme is used (see details of the algorithm in [21]). Additionally, the motion model used for SIS-based tracking is exactly that designed for the proposed method, while KF-based tracking uses the same dynamic

model for independent motion of vehicles, but cannot accommodate any interaction model. Other parameters of the dynamic model are $w_l = 90$ and $d_s = 96$. Regarding the observation model, the dimensions of the local windows R_a and R_b are set to $w = h = 10$. Also, the standard deviation of the proposal density is optimally calculated for the proposed method and for SIS-based tracking as $\sigma_q = (2, 3)$ and $\sigma_q = (5, 8)$, respectively. Finally, the same number of samples $N = 250$ is used for both methods.

To compare methods, the number of tracking failures incurred by each of the methods on the same test sequences is counted. A tracking failure occurs when the tracker fails to provide continuous and coherent measures for a given vehicle inside the region of interest (ROI). The ROI is defined to be the scope of the IPM, which usually comprises the own and the two adjacent lanes, and extends longitudinally up to a distance d_f that depends on the camera calibration. Comparative results are displayed in Table 2. As expected, the proposed method largely outperforms the others in terms of tracking failures in the test sequences. Naturally, KF-based tracking delivers the highest error rate as it is unable to handle situations in which vehicles interact. Notably, SIS-based tracking also outputs a significant number of errors, since the number of particles is relatively small and these fail to correctly sample the space when the number of vehicles grows.

7.3 Analysis of computational complexity

The computational cost of the different processing blocks of the method is summarized in Table 3. As can be observed, the appearance analysis is naturally the most costly module (44% of the processing time per frame) as it constitutes the core for the definition of the observation model and is in the basis of the hypotheses generation for vehicle detection. Most remarkably, the vehicle tracking algorithm consumes only 23.98% of the processing time thanks to the efficient MCMC sampling.

Indeed, compared to traditional particle filtering based on importance sampling, the processing load of sampling is largely lightened. In MCMC, for a given number of objects M, a Markov chain of length $C \cdot M$ is created, in which the state of each object changes on average C times (in each step of the chain only one random object is moved). Hence, samples generated from the chain may comprise a number of possible object state combinations at the order of C^M. In contrast, in SIS-based sampling, particles are independently generated and each corresponds to a possible state combination, therefore C^M particles are necessary to obtain the same number of combinations as in MCMC. Consequently, since each particle involves evaluation of the posterior density in (11) both for MCMC and SIS, the complexity of the proposed sampling algorithm is $\Omega(C \cdot M)$ whereas that of SIS-based sampling is $\Omega(C^M)$. The Big Omega Ω is used here to denote the lower bound, since a number of lighter additional operations are also involved in sampling besides posterior density evaluation. In other words, the complex-

ity of the sampling algorithm grows linearly in the proposed method, while the traditional SIS grows exponentially. Complexity comparison between methods is summarized in Table 4.

The real-time operation requirement constrains the processing time available for particle propagation and thus the efficiency of the sampling is crucial to the performance of the method. As shown in Table 5, if we limit the number of particles to $N = 250$, for which near real-time operation is attained (approximately 10 fps), SIS-based tracking delivers 31 tracking failures in the test sequences described above, whereas with the same computational resources the proposed method outputs only 9 failures. Even if the number of particles for SIS-based tracking is increased to $N = 1000$, the tracking failures still outnumber largely those of the proposed method, while the average frame rate plummets far from real-time. As a matter of fact, if we consider a mean number of $M = 2$ observed vehicles per image, traditional sampling would theoretically require approximately $(250/2)^2 = 15625$ particles to achieve a performance similar to the proposed method. As a final remark, the use of the interaction model explained in Section 4 increases the processing time per frame from 94.78 ms to 96.06 ms and thus entails an overhead of only 1.34% which is well compensated by the gain in performance.

7.4 Discussion and examples

The strength of the method lies to a great extent in the combination of two different sources of information (appearance and motion) for the definition of the observation model. Indeed, the combination of information ensures that whenever the two sources are available a robust average estimate is produced, and most importantly, it allows for the tracking of the objects even if one of the information sources is unavailable or unreliable. Fig. 10.a illustrates the sampling process for the original image in Fig. 10.a1 whenever the two types of information are available. In particular, Fig. 10.a2 shows the rectified domain after IPM application, Fig. 10.a3 corresponds to the appearance-based pixel-wise classification (in which pixels likely belonging to the lower parts of vehicles are colored in white as explained in Section 5.1.2), and Fig. 10.a4 contains analogously the pixel-wise motion-based classification as explained in Section 5.2.2.

The process of sample generation in the framework of the Markov chain is superimposed on Fig. 10.a3 and Fig. 10.a4. In particular, the segment between the previous sample and the proposed sample is colored in green whenever the latter is accepted and in red if it is rejected. As can be observed, accepted samples concentrate in the area of high likelihood (i.e., the transition between road and vehicles), while samples diverging from this area are rejected. The final estimates for vehicles positions are indicated in Fig. 10.a5 with white segments underlining the vehicle rears.

As stated, dual modeling from two sources prevents the method from losing track whenever one of the sources is unreliable. This is illustrated in Fig. 10.b, and Fig. 10.c, where the sampling process is depicted analogously to Fig. 10.a for the images in Fig. 10.b1 and Fig. 10.c1. In particular, in Fig. 10.b the motion-based observation provides no measurement for the right vehicle (Fig. 10.b4), however this is compensated by the correct appearance-based observation, which avoids particle dispersion. Therefore, the vehicle is correctly tracked as shown in Fig. 10.b5. In contrast, the particles for the left vehicle overcome an inaccurate initialization and converge to its actual position due to good appearance-based and motion-based observations. The opposite case is illustrated in Fig. 10.c, in which the appearance-based model fails to detect the furthest vehicle (see Fig. 10.c3), whereas the region of motion is still observable in the difference between aligned images in Fig. 10.c4. The combinations of the two sources results in the correct tracking of the vehicle, as shown in Fig. 10.c5.

Finally some graphical examples of the performance of the method are shown in Fig. 11.. This figure displays snapshots of the tracking process for four different time points (from left to right) in three different example sequences. In the first sequence, the method simultaneously tracks a vehicle that is being rapidly overtaken by the own vehicle. Most interestingly, there is some degree of interaction between the vehicles, in fact, they are fairly close in Fig. 11.a3. In traditional SIS-based tracking methods, particles are prone to concentrate around the object with the highest likelihood which

results in the loss of the other object. In contrast, the designed interaction model allows for the prevention of this situation and the successful tracking of the vehicles until they part. In the second example, in Fig. 11.b, tracking of a vehicle driving at a slow pace in the right lane is shown. At the same time, the method swiftly detects a vehicle in the left hand side (Fig. 11.b3) and tracks it until it is far away, while also keeping track of the vehicle in front of the own car. Finally, in Fig. 11.c simultaneous tracking of several vehicles is shown: the vehicle ahead the own car moves from the lower-left corner of the image to the upper-middle part, while at the same time tracking is kept for the distant vehicle in the right lane. Meanwhile, a new vehicle entering the scene in the left hand is detected and tracked until it is nearly at the same distance as the other vehicles.

8 Conclusions

In this article, a new probabilistic framework for vehicle detection and tracking has been presented based on MCMC. As regards vehicle detection, a new descriptor, CR-HOG, has been defined based on the extraction of gradient features in radial rectangular bins. The descriptor has proven to have good discriminative properties using a reduced number of features in a simple linear-kernel SVM classifier, and is thus ideally suited for real-time applications. In addition, the tracker is proven to perform better than state-of-the-art methods based on Kalman and particle filtering in terms of tracking failures. The power of the algorithm lies in the fusion of information of dif-

ferent nature, especially regarding the observation model. In effect, aside from appearance the method incorporates the analysis of another feature that is inherent to vehicles: their motion. In addition, MCMC method is capitalized on to perform efficient sampling and to avoid the performance degradation of particle filter-based approaches in multiple object tracking arising from the curse of dimensionality. In summary, the method is able to overcome the common limitations of particle filter-based approaches and to provide robust vehicle tracking in a wide variety of driving situations and environment conditions.

Competing interests

The authors declare that they have no competing interests.

Authors' contributions

JA carried out the design of the vehicle tracking algorithm, the dynamic model, the motion-based observation model, and the vehicle detection algorithm. LS conceived of the study and participated in its design and coordination. MN carried out the design of the appearance-based observation model. All authors read and approved the final manuscript.

Acknowledgment

This study was supported in part by the Ministerio de Ciencia e Innovación of the Spanish Government under projects TEC2007-67764 (SmartVision) and TEC2010-20412 (Enhanced 3DTV).

References

1. GY Song, KY Lee, JW Lee, in *Proceedings of the IEEE Intelligent Vehicles Symposium*, Eindhoven, The Netherlands, 4–6 June 2008
2. L Gao, C Li, T Fang, Z Xiong, in *Image Analysis and Recognition*, ed. by A Campilho, M Kamel. *International Conference on Image Analysis and Recognition*, Póvoa de Varzim, June 2008. Lecture Notes in Computer Science, vol. **5112** (Springer, Heidelberg, 2008), pp. 142–150
3. L-W Tsai, J-W Hsieh, K-C Fan, IEEE Trans. Image Process. **16**(3), 850–864 (2007)
4. W Liu, X Wen, B Duan, H Yuan, N Wang, in *Proceedings of the IEEE Intelligent Vehicles Symposium*, Istanbul, Turkey, 13–15 June 2007
5. A Ess, B Leibe, K Schindler, L van Gool, IEEE Trans. Pattern Anal. Mach. Intell. **31**(10), 1831–1846 (2009)
6. G Toulminet, M Bertozzi, S Mousset, A Bensrhair, A Broggi, IEEE Trans. Image Process. **15**(8), 2364–2375 (2006)
7. TN Tan, IEEE Trans. Image Process. **9**(8), 1343–1356 (2000)
8. JM Collado, C Hilario, A de la Escalera, JM Armingol, in *Proceedings of the IEEE Intelligent Vehicles Symposium*, University of Parma, Italy, 17 June 2004

9. DM Ha, J-M Lee, Y-D Kim, Image Vis. Comput. **22**, 899–907 (2004)
10. Z Sun, G Bebis, R Miller, in *Proceedings of the International Conference on Digital Signal Processing*, Santorini, Greece, 1–3 July 2002
11. Z Sun, G Bebis, R Miller, IEEE Trans. Image Process. **15**(7), 2019–2034 (2006)
12. N Dalal, B Triggs, in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, San Diego, California, 20–26 June 2005
13. P Negri, X Clady, SM Hanif, L Prevost, A cascade of boosted generative and discriminative classifiers for vehicle detection. EURASIP J. Adv. Signal Process. (2008). doi:10.1155/2008/782432
14. C-CR Wang, J-JJ Lien, IEEE Trans. Intell. Transport. Syst. **9**(1), 83–96 (2008)
15. C Papageorgiou, T Poggio, Int. J. Comput. Vis. **38**(1), 15–33 (2000)
16. R Lienhart, J Maydt, in *Proceedings of the IEEE International Conference on Image Processing*, Rochester, New York, 22–25 September 2002
17. J Arrospide, L Salgado, M Nieto, F Jaureguizar, in *Proceedings of the IEEE International Conference on Image Processing*, San Diego, California, 12–15 October 2008
18. Y Chen, M Das, D Bajpai, in *Proceedings of the Canadian Conference on Computer and Robot Vision*, Montreal, Quebec, 28–30 May 2007
19. R Goecke, N Pettersson, L Petersson, in *Proceedings of the IEEE Intelligent Vehicles Symposium*, Istanbul, Turkey, 13–15 June 2007
20. M Asadi, F Monti, CS Regazzoni, Feature classification for robust shape-based collaborative tracking and model updating. EURASIP J. Image Video Process. (2008). doi:10.1155/2008/274349

21. MS Arulampalam, S Maskell, N Gordon, T Clapp, IEEE Trans. Signal Process. **50**(2), 174–188 (2002)

22. A Dore, M Soto, CS Regazzoni, IEEE Signal Process. Mag. **27**(5), 46–55 (2010)

23. JJ Pantrigo, A Sánchez, AS Montemayor, A Duarte, Pattern Recogn. Lett. **29**, 1160–1174 (2008)

24. T Gao, Z-G Liu, W-C Gao, J. Zhang, in *Advances in Neuro-Information Processing*, ed. by M Köppen, N Kasabov, G Coghill. *15th International Conference on Neural Information Processing of the Asia-Pacific Neural Network Assembly*, Auckland, Nov. 2008. Lecture Notes in Computer Science, vol. **5507** (Springer Berlin, Heidelberg, 2008), pp. 695–702

25. C Luo, X Cai, J Zhang, in *Proceedings of IEEE Workshop on Multimedia Signal Processing*, Cairns, Australia, 8–10 October 2008

26. J Wang, Y Ma, C Li, H Wang, J Liu, in *Proceedings of World Congress on Computer Science and Information Engineering*, Los Angeles, California, 31 March–2 April 2009

27. Z Khan, T Balch, F Dellaert, IEEE Trans. Pattern Anal. Mach. Intell. **27**(11), 1805–1819 (2005)

28. J Wang, Y Yin, H Man, Multiple human tracking using particle filter with Gaussian process dynamical model. EURASIP J. Image Video Process. (2008). doi:10.1155/2008/969456

29. K Smith, D Gatica-Perez, J-M Odobez, in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, San Diego, California, 20–26 June 2005

30. B-N Vo, B-T Vo, N-T Pham, D Suter, IEEE Trans. Signal Process. **58**(10), 5129–5141 (2010)

31. CM Bishop, *Pattern Recognition and Machine Learning* (Springer, New York, 2006)
32. SK Pang, J Li, SJ Godsill, IEEE Trans. Aerosp. Electron. Syst. **47**(1), 472–502 (2005)
33. M Bertozzi, A Broggi, Comput. Vis. Image Understand. **113**(6), 743–749 (1998)
34. R Danescu, S Nedevschi, M-M Meinecke, T-B To, in *Proceedings of the IEEE Intelligent Vehicles Symposium*, Eindhoven, The Netherlands, 4–6 June 2008
35. M Nieto, L Salgado, F Jaureguizar, J Cabrera, in *Proceedings of the IEEE Intelligent Vehicles Symposium*, Istanbul, Turkey, 13–15 June 2007
36. JA Bilmes, A Gentle Tutorial of the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models, Technical Report TR-97-021 (1998)
37. RI Hartley, A Zisserman, *Multiple View Geometry in Computer Vision* (Cambridge University Press, Cambridge, 2000)
38. TK Moon, WC Stirling, *Mathematical Methods and Algorithms for Signal Processing* (Prentince Hall, Englewood Cliffs, NJ, 1999)
39. T Zielke, T Brauckmann, W Von Seelen, CVGIP: Image Understand. **58**(2), 177–190 (1993)
40. The GTI-UPM Vehicle Image Database, http://www.gti.ssr.upm.es/data
41. The Caltech Database (Computational Vision at California Institute of Technology, Pasadena), http://www.vision.caltech.edu/html-files/archive.html. Accessed 14 May 2011
42. R Fergus, P Perona, A Zisserman, in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Madison, Wisconsin, 16–22 June 2003

43. The TU Graz-02 Database (Graz University of Technology), http://www.emt.tugraz.at/~pinz/data/GRAZ_02/. Accessed 14 May 2011

44. A Opelt, A Pinz, in *Proceedings of the 14th Scandinavian Conference on Image Analysis*, Joensuu, Finland, 19–22 June 2005

Fig. 1. General scheme of the proposed method.

Fig. 2. Transformation to the rectified domain through IPM. As opposed to the original image **(a)**, in the rectified image **(b)** the effect of perspective is removed and thus motion of vehicles is easier to model.

Fig. 3. Relative pose of the camera at two different time points k_1 and k_2. The world coordinate system has its origin at the position of the camera center at k_1.

Fig. 4. Example of image alignment. Image **(a)** and **(b)** correspond to times $k-1$ and k, respectively, of the video sequence; **(c)** is the image at $k-1$ warped with $\hat{H}^k$; **(d)** is the difference between aligned images, i.e., **(b)** and **(c)**; and **(e)** is the corresponding image of differences in the rectified domain. Both **(d)** and **(e)** have been binarized for better visualization: white regions correspond to regions of difference, which usually correspond to the lower edge of vehicles.

Fig. 5. Example of generation of a new vehicle hypothesis. The sequence of images is the following: **(a)** original image, **(b)** rectified image, **(c)** binary map B_m corresponding to appearance analysis in (b): pixels in white indicate potential location of vehicles. In the example, the regions labeled 1 and 2 correspond to existing vehicles, while the small region arising in the lower left corner constitutes a potential new vehicle.

Fig. 6. Combined HOG and symmetry based descriptor. **(a)** The structure of concentric rectangle HOG (CR-HOG) with its corresponding parameters is illustrated. **(b)** The refined regions obtained after vertical symmetry analysis is shown for some examples: green and red lines indicate respectively the symmetry axis and the width of the region yielding the maximum symmetry values.

Fig. 7. Possible configurations of CR-HOG regarding the number of orientation bins. The range of gradient orientation angles [0–180) is divided in uniformly spaced sectors. Pixels with gradient orientations inside each sector accumulate to the corresponding bin of the histogram proportionally to the magnitude of their gradient. Configurations with **(a)** 8, **(b)** 12, and **(c)** 18 bins are considered.

Fig. 8. Classification accuracy as a function of the number of cells, b. The results are broken down for images corresponding to **(a)** close/middle, **(b)** left, **(c)** right and **(d)** far views.

Fig. 9. Classification accuracy as a function of the number of orientation bins, n. The results are broken down by zones: **(a)** close/middle, **(b)** left, **(c)** right and **(d)** far.

Fig. 10. Illustration of sampling process for different example images. From left to right, images correspond to the (1) original image, (2) rectified domain, (3) appearance-based vehicle probability map, B_m, (4) motion-based vehicle probability map, and (5) tracking results. The sampling process is illustrated in images (3) and (4): accepted and rejected particles are painted in green and red, respectively. Images (2)–(4) are zoomed for better visualization of the sampling process. Images in **(a)** illustrate a normal sampling scenario, while images in **(b)** and **(c)** show how combined sampling is able to overcome bad **(b)** motion-based and **(c)** appearance-based measurements.

Fig. 11. Vehicle tracking for three different sequences (a)–(c). From left to right, the images show results at times $k_0, k_0 + 200, k_0 + 340, k_0 + 440; k_0, k_0 + 170, k_0 + 215, k_0 + 295; k_0, k_0 + 250, k_0 + 360, k_0, k_0 + 460$ for sequences (a), (b), and (c), respectively.

Table 1 Classification accuracy rates of CR-HOG

	Close/Middle			**Left**			**Right**			**Far**		
	$b=2$	$b=3$	$b=4$	$b=2$	$b=3$	$b=4$	$b=2$	$b=3$	$b=4$	$b=2$	$b=3$	$b=4$
$n=8$	94,88	94,98	94,68	91,04	91,18	91,16	88,58	89,14	87,94	85,92	85,86	85,76
$n=12$	94,96	94,80	95,14	91,46	91,82	91,46	89,28	89,42	88,16	85,32	85,24	85,16
$n=18$	94,78	93,96	93,24	91,98	91,60	91,06	89,34	88,84	88,10	85,76	85,22	84,60

Table 2 Summary of tracking results

Method	**Tracking failures**	**Number of frames**	**Number of vehicles**
KF-based Tracking	36		
SIS-based Tracking	31	33454	120
Proposed Method	9		

Table 3 Average processing time of the proposed algorithm per block

	Appearance analysis	**Motion analysis**	**Vehicle detection**	**Sampling**	**Total**
Processing time (ms)	44.33	9.15	19.54	23.04	96.06
Processing load (%)	46.15	9.52	20.35	23.98	100

Table 4 Comparison of time complexity between SIS-based tracking and the proposed method

	SIS-based tracking	**Proposed method**
Number of vehicles	M	M
Number of particles	C^M	$C \cdot M$
Time complexity	$\Omega(C^M)$	$\Omega(C \cdot M)$

Table 5 Performance comparison between SIS-based tracking and the proposed method

Method	Number of particles	Processing time for sampling (ms)	Average time per frame (ms)	Frame processing rate (fps)	Tracking failures
SIS-based tracking	250	23.55	96.56	10.36	31
SIS-based tracking	1000	114.33	187.34	5.34	22
Proposed method	250	23.04	96.06	10.41	9

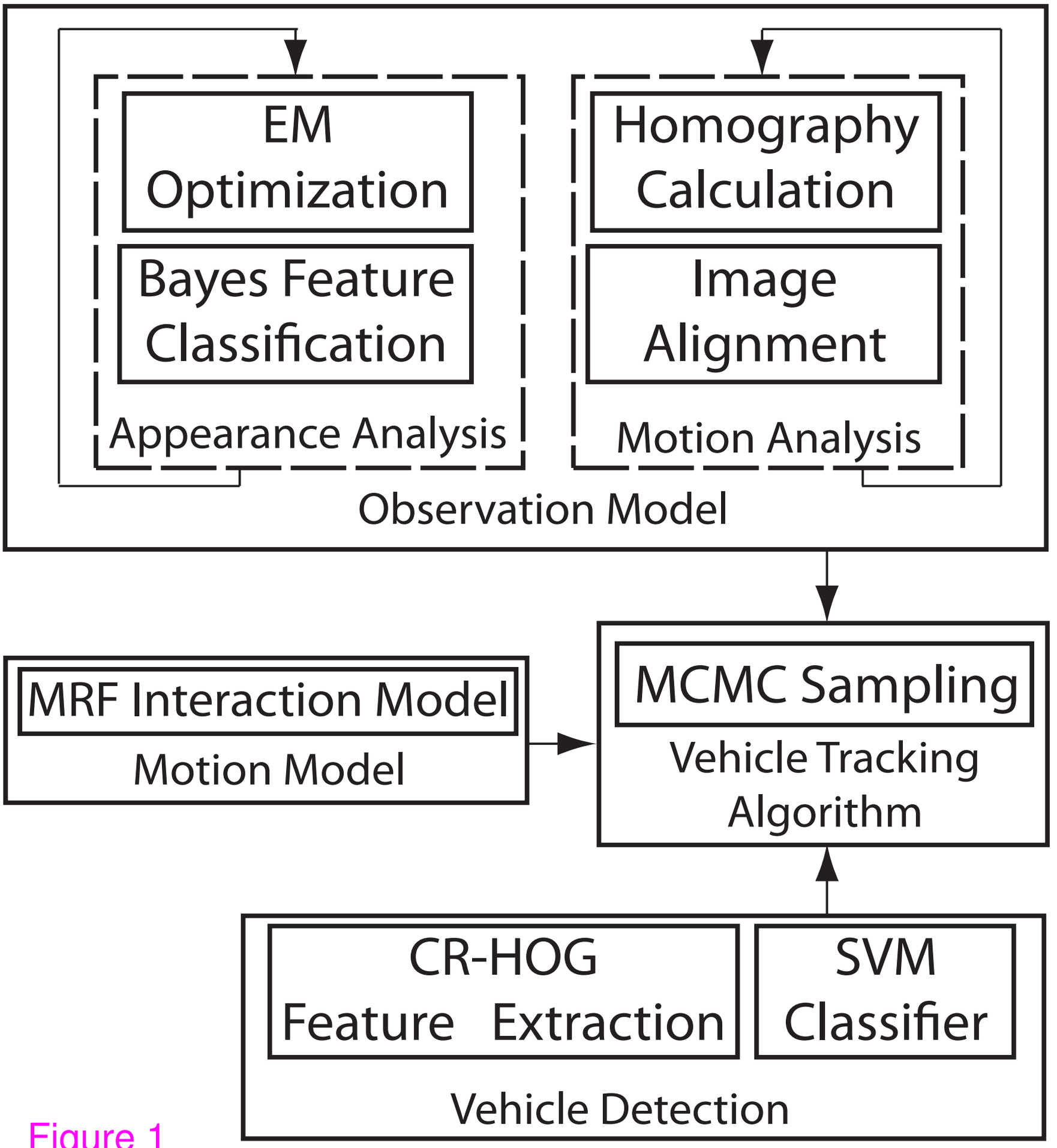

EM Optimization
Bayes Feature Classification
Appearance Analysis
Homography Calculation
Image Alignment
Motion Analysis
Observation Model
MRF Interaction Model
Motion Model
MCMC Sampling
Vehicle Tracking Algorithm
CR-HOG Feature Extraction
SVM Classifier
Vehicle Detection

Figure 1

Figure 2 (a) (b)

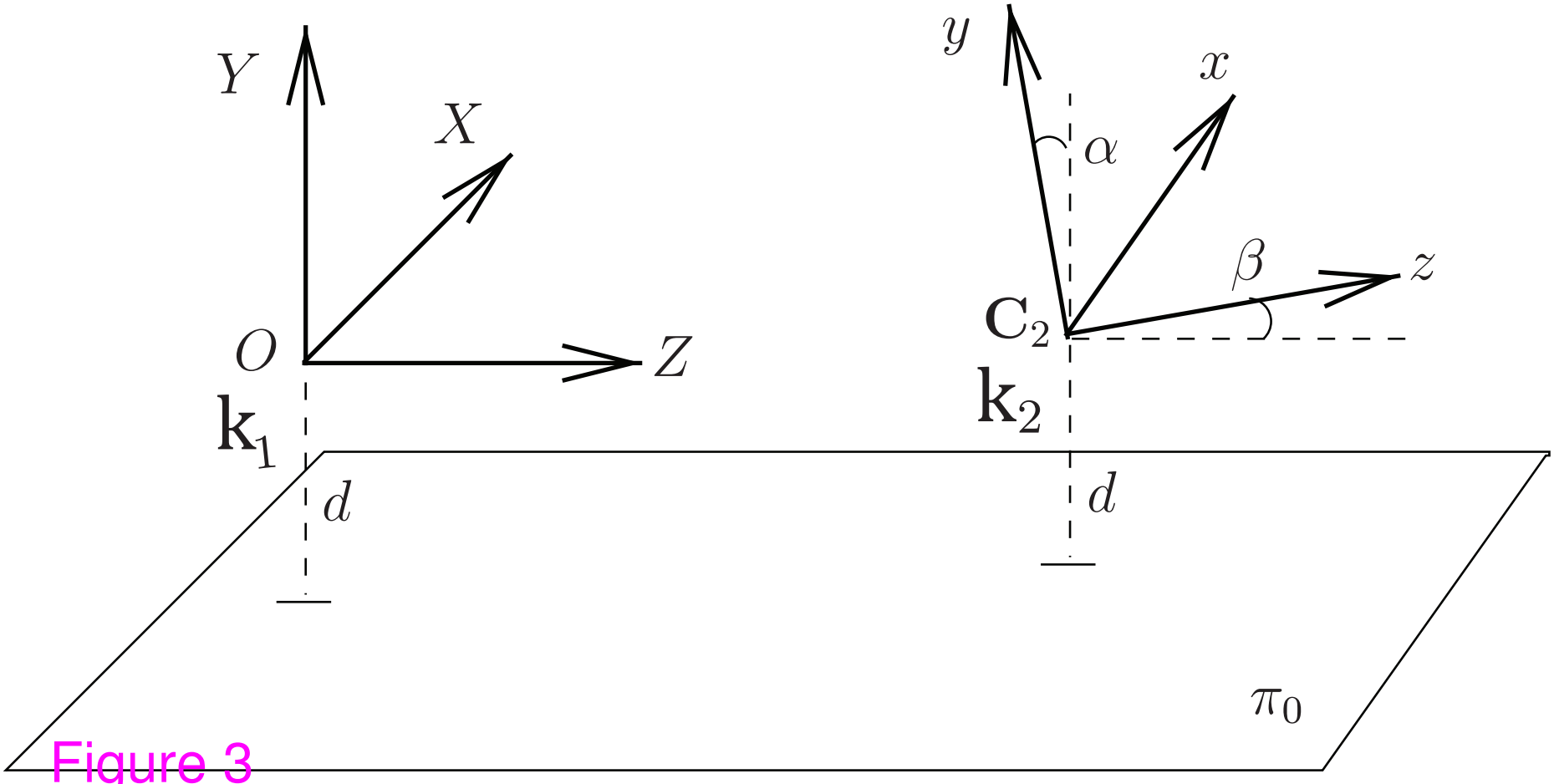


Figure 3

Figure 4 (a) (b) (c) (d) (e)

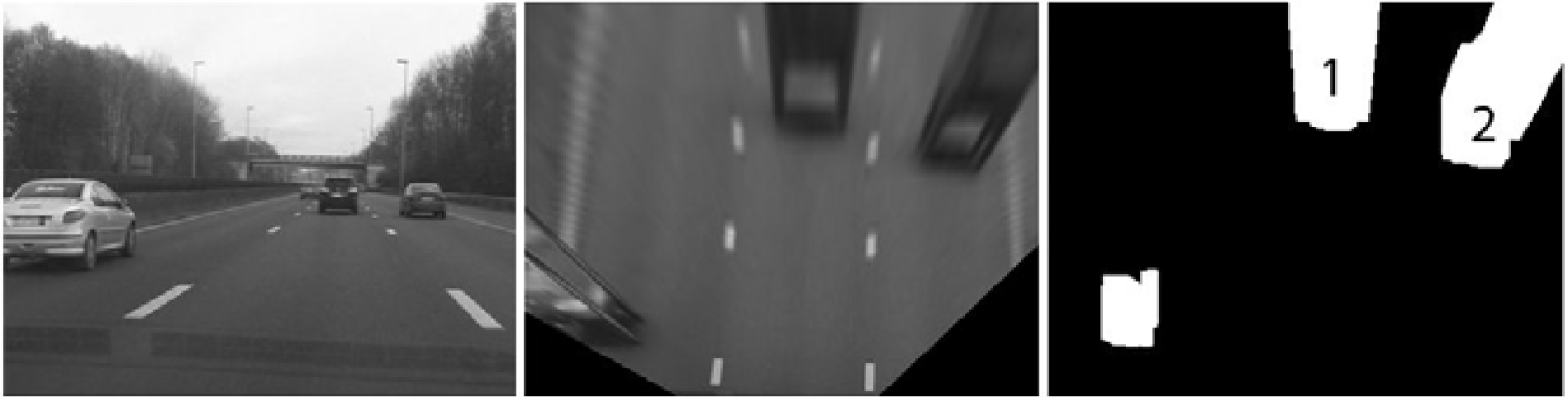


Figure 5 (a) (b) (c)

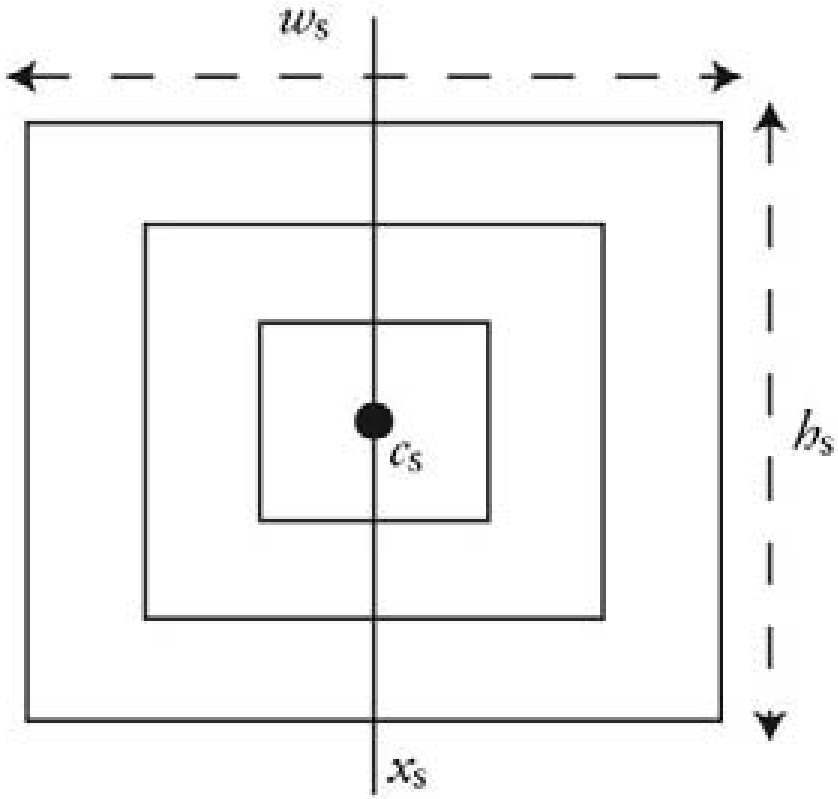
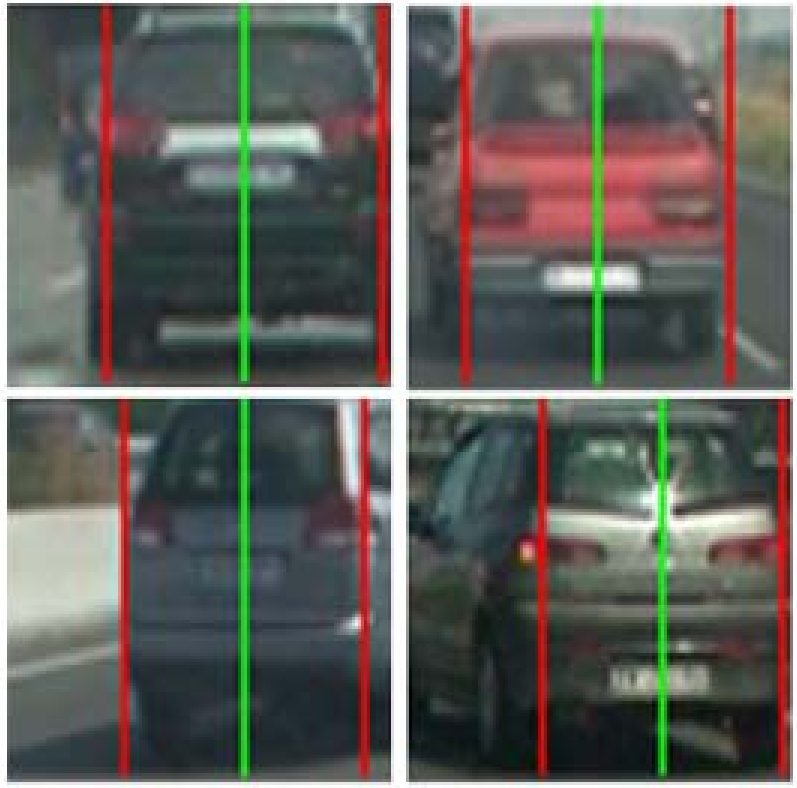
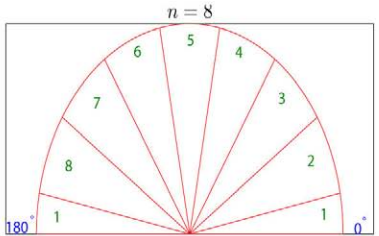


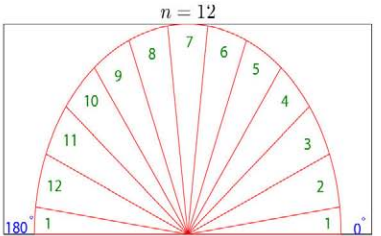
Figure 6(a)



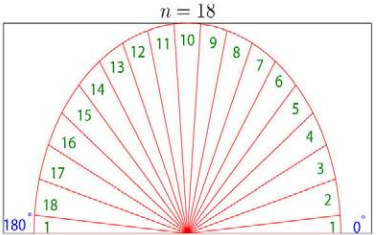
(b)


$n = 8$
1
2
3
4
5
6
7
8
180°
0°

(a)

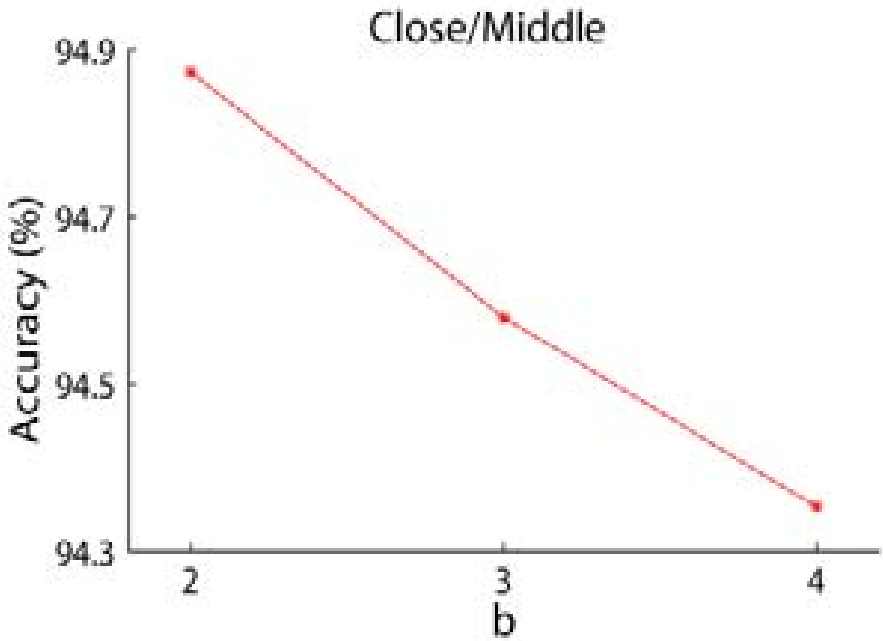

$n = 12$
1
2
3
4
5
6
7
8
9
10
11
12
180°
0°

(b)

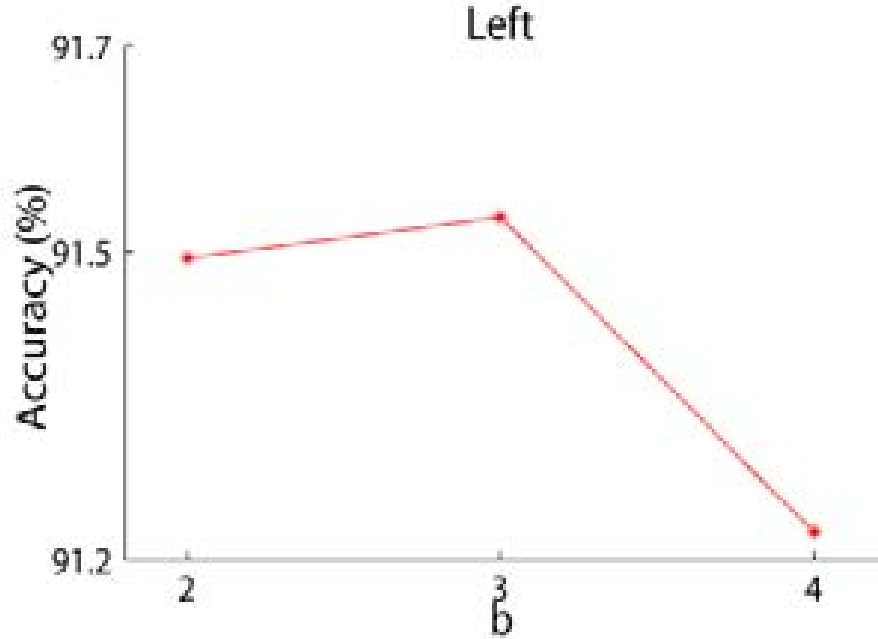

$n = 18$
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
180°
0°

(c)

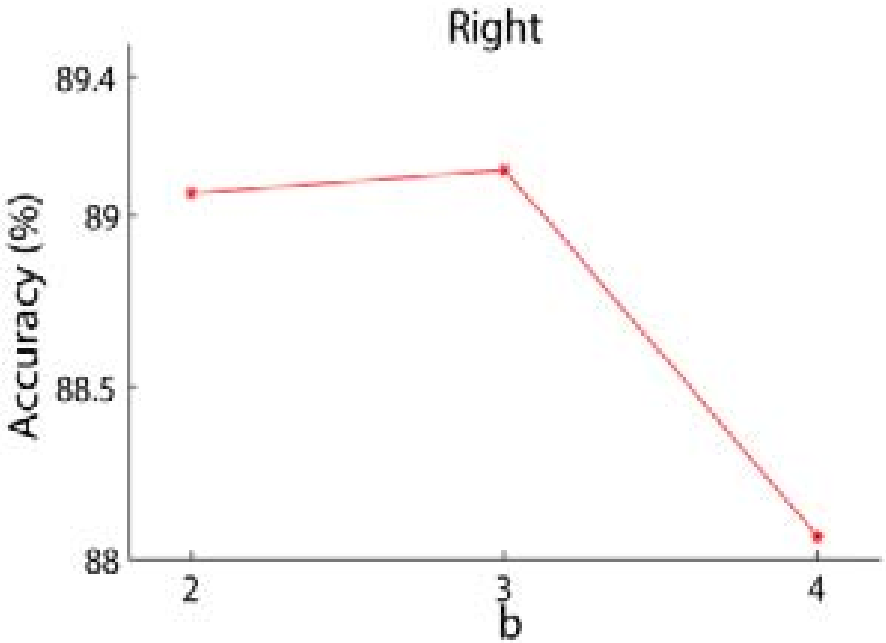
Figure 7



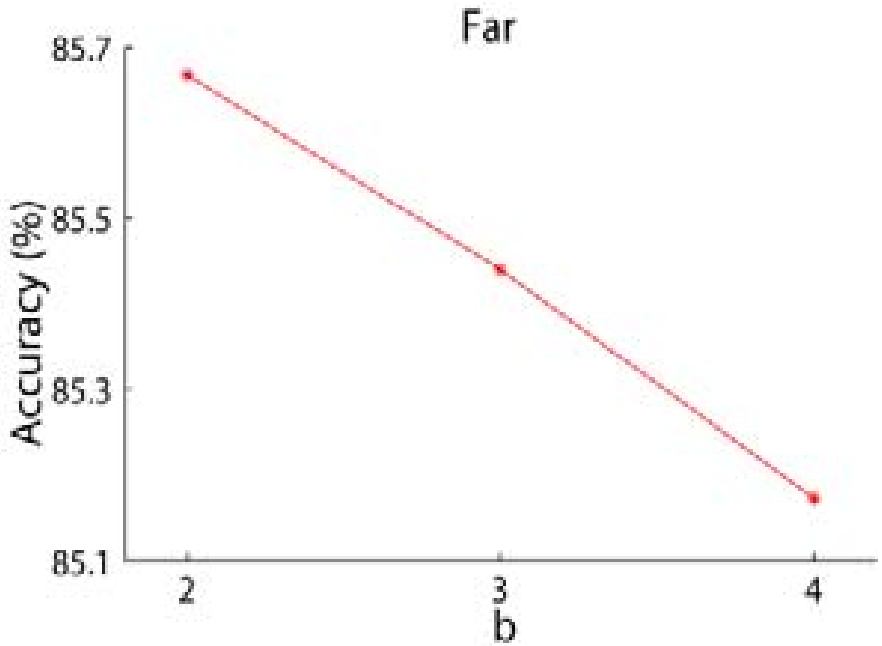
(a)



(b)

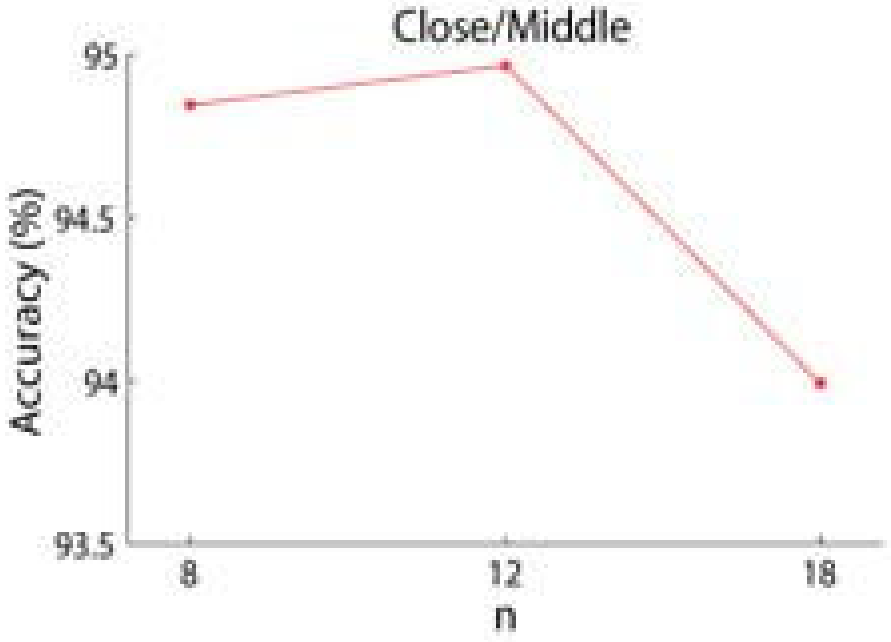


(c)

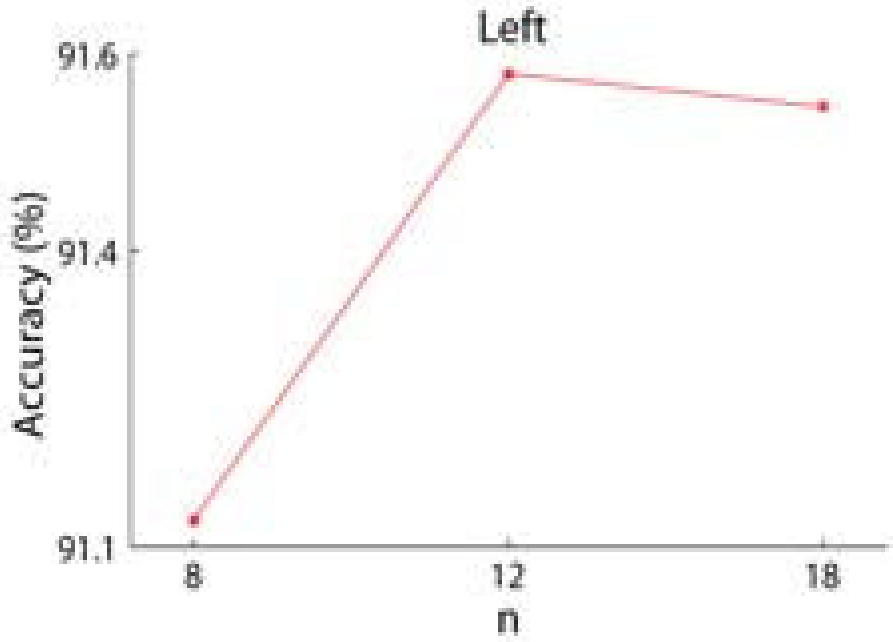


(d)

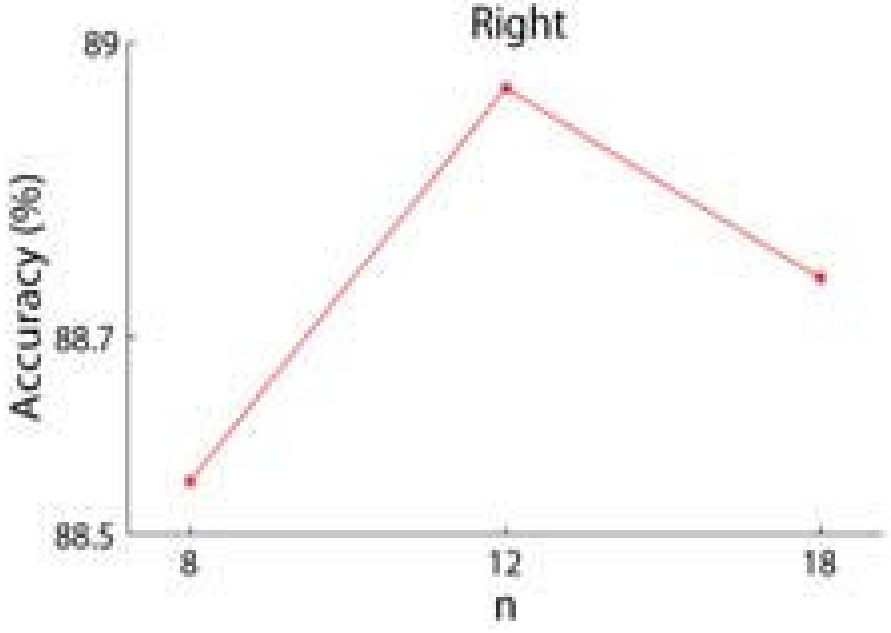
Figure 8



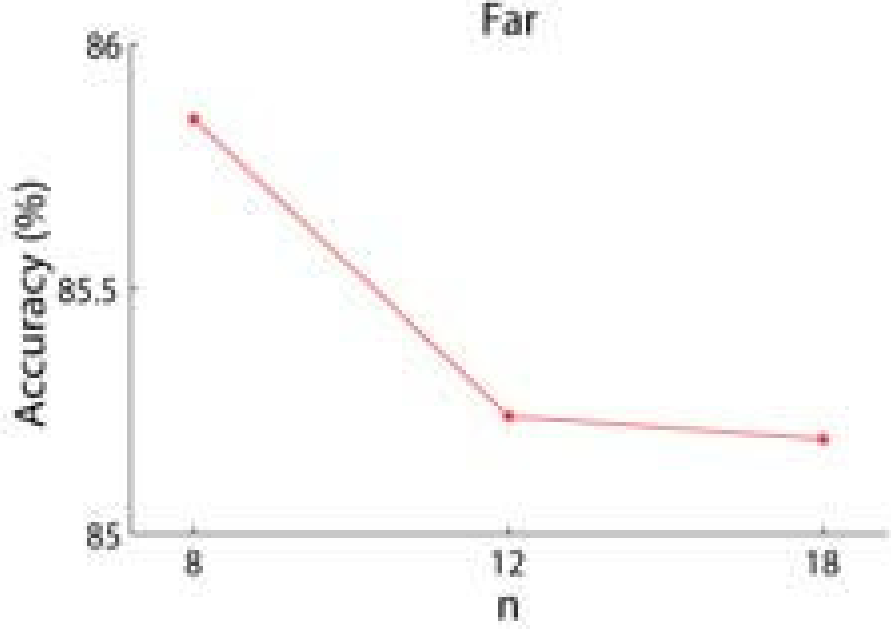
(a)



(b)

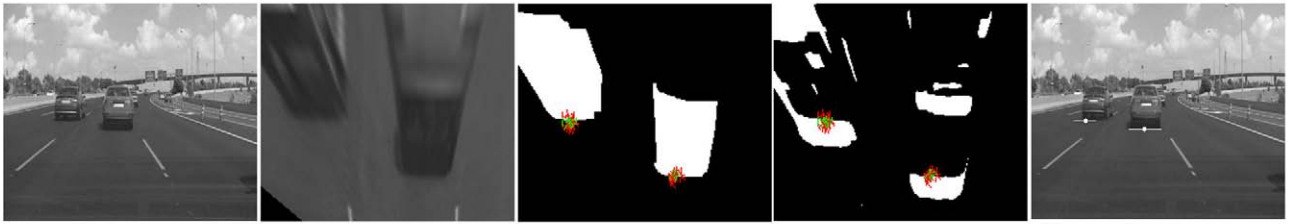


(c)



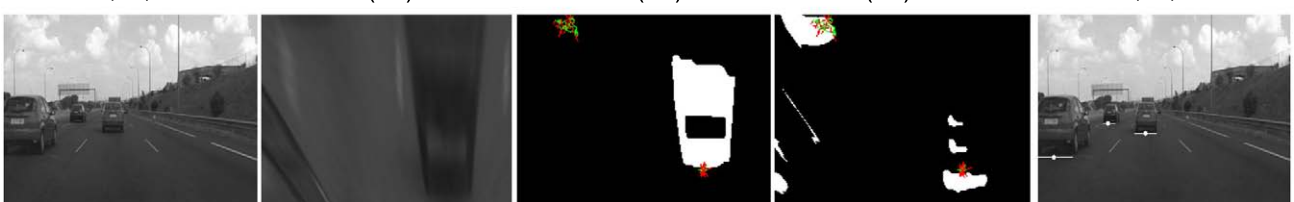
(d)

Figure 9



(a1) (a2) (a3) (a4) (a5)

(b1) (b2) (b3) (b4) (b5)



(c1) (c2) (c3) (c4) (c5)

Figure 10

(a1) (a2) (a3) (a4)

(b1) (b2) (b3) (b4)

(c1) (c2) (c3) (c4)

Figure 11