J Real-Time Image Proc
DOI 10.1007/s11554-012-0315-0

ORIGINAL RESEARCH PAPER

# Real-time lane tracking using Rao-Blackwellized particle filter

**Marcos Nieto · Andoni Cortés · Oihana Otaegui · Jon Arróspide · Luis Salgado**

Received: 28 June 2012 / Accepted: 12 December 2012

**Abstract** A novel approach to real-time lane modeling using a single camera is proposed. The proposed method is based on an efficient design and implementation of a particle filter which applies the concepts of the Rao-Blackwellized particle filter (RBPF) by separating the state into linear and non-linear parts. As a result the dimensionality of the problem is reduced, which allows the system to perform in real-time in embedded systems. The method is used to determine the position of the vehicle inside its own lane and the curvature of the road ahead to enhance the performance of advanced driver assistance systems. The effectiveness of the method has been demonstrated implementing a prototype and testing its performance empirically on road sequences with different illumination conditions (day and nightime), pavement types, traffic density, etc. Results show that our proposal is capable of accurately determining if the vehicle is approaching the lane markings (Lane Departure Warning), and the curvature of the road ahead, achieving processing times below 2 ms per frame for laptop CPUs, and 12 ms for embedded CPUs.

M. Nieto (✉) · A. Cortés · O. Otaegui
Vicomtech-ik4, Paseo Mikeletegi 57,
Donostia–San Sebastián, Spain
e-mail: mnieto@vicomtech.org

A. Cortés
e-mail: acortes@vicomtech.org

O. Otaegui
e-mail: ootaegui@vicomtech.org

J. Arróspide · L. Salgado
Grupo de Tratamiento de Imágenes, Universidad Politécnica de Madrid, Madrid, Spain
e-mail: jal@gti.ssr.upm.es

L. Salgado
e-mail: lsa@gti.ssr.upm.es

## 1 Introduction

Currently, existing advanced driver assistance systems (ADAS) based on time-of-flight sensors (radar) are able to accurately determine the presence, distance and speed of objects in front of a vehicle. This feat allows services such as safety-distance warning, stop-and-go, etc [12]. This type of systems are specially interesting for long-distance buses, where the safety is critical and the cost of such devices is affordable for manufacturers. However, these systems do not take into account the curvature of the road, therefore many false alarms are generated when a vehicle (or any static element of the scenario, such as guardrails) is physically close, but in a different lane.

Vision-based systems can handle this information and incorporate it into ADAS, potentially reducing the number of false alarms [13]. Since the seminal works of Dickmanns [8] during the 80's, the literature has evolved and presented a number of studies proposing curvature models ranging from parabolic models [12] to complex splines [16] or clothoid models [5]. Last trends have put the effort in the definition of probabilistic methods (e.g. particle filters with sequential importance sampling or Markov Chain Monte Carlo methods, MCMC, sampling) that, combined with a previous calibration stage, achieve high degrees of fitting accuracy, using both mono [2, 11] and stereo approaches [3, 4, 6].

Nevertheless, these road modeling methods have some important drawbacks. First, they require the computation of hundreds or thousands of hypotheses [15], which, in the case of particle filters methods, imply the computation of

Published online: 27 December 2012

too many point-wise estimates of the posterior density distribution. Consequently, processing requirements might exceed the computational power of embedded processors. Second, these methods typically need the calibration of the camera, which is also a source of inaccuracies and causes practical additional difficulties (especially for stereo approaches). And finally, most of them make use of complex methods such as inverse perspective mapping that cannot be easily migrated to embedded platforms.

In this work, we have focused on obtaining a robust road model, which provides time-coherent estimations of the position of the vehicle inside the lane and the curvature of the road ahead, without the need to compute a calibration stage.

Analogously to [6, 11], the proposed method is based on the Bayesian inference theory, since it provides an unbeatable framework to combine different nature information. Nevertheless, we propose several enhancements devoted to reduce its associated computational cost to perform in real-time in embedded processors. Our main contribution is the separation of the linear and non-linear part of the road model to apply the concepts of Rao-Blackwellized particle filters (RBPF) [7, 9].

Our methodology dramatically reduces the number of samples the particle filter uses in comparison with traditional sampling schemes. The main reason is that the reduction of the dimensionality of the state vector implies an exponential reduction of the amount of samples in an importance sampling scheme [10, 14].

The rest of the paper is structured as follows: Sect. 2 introduces the block diagram of the proposed method and Sect. 3 describes the mathematical model used to characterize the road. Section 4 presents the probabilistic framework that is divided into linear and non-linear components, which are described in Sects. 5 and 6, respectively. The last two sections deepen in the computational cost of the proposed method and the obtained results. Specifically, execution times below 12 and 2 ms per frame are respectively obtained in an embedded processor and in a desktop CPUs. The proposed solution has been tested in sequences acquired on buses and cars driven in highways, with varied illumination conditions, number of vehicles, lane markings visibility, etc, where it has proven to yield excellent results.

## 2 System overview

The proposed method receives as input the sequence of images captured with a forward-looking single camera and generates as output a road model at each point in time. The latter can be used as well to connect the proposed method to other ADAS modules, such as the Human Machine Interface (HMI), the CAN bus, obstacle detection sensors, etc.

The workflow inside the video processing module is shown in Fig. 1. As shown, the process can be basically divided in two pipelines, for the linear part and non-linear part of the computations. The former chain first detects lane markings in the images, which are then accumulated using a perspective LUT histogram. The search for maxima leads to the generation of linear measurements of the state vector that are fed into the Kalman filter. The non-linear part takes as input both the lane markings detections and the result of the linear part, and using a particle filter generates the estimates of the remaining variables of the model.

Next section introduces the mathematical road model, and how it is parameterized.

## 3 Road model

A multiple-lane model composed of the center lane and the adjacent left and right lanes is proposed. The road is assumed to be locally planar (as done by many authors [11, 12], and each lane consists of two lane markings, which are described as curves on the road plane (the curvature is assumed to be the same for all lanes).

The image plane is rectified using a planar homography to give a bird's-eye view that can be estimated automatically by calculating the vanishing point of the road lines [13]. Figure 2a, b shows an example image and its corresponding rectified view.

In this bird's-eye view, we have observed that a circumference model provides enough fitting accuracy. As a general criticism to other, more complex curve models, the gain in accuracy is usually constrained by the quality of the observations, so that in this kind of scenarios, where there is typically a significant perspective effect and a lack of reliable information at far distance, an increase in the complexity of the curvature model results in no additional accuracy.

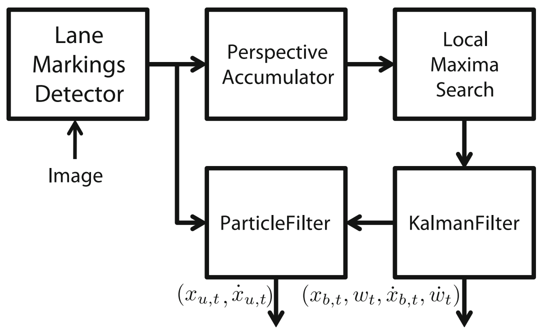


**Fig. 1** Block diagram of the proposed video processing algorithm

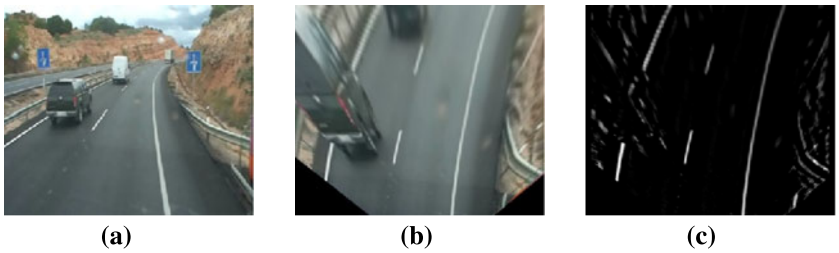


**Fig. 2** Road image (**a**), its bird's-eye view (**b**) and the detected lane markings (**c**) using the method proposed in [13]

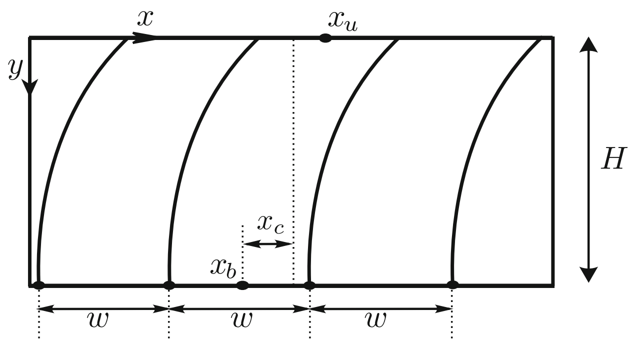


**Fig. 3** Example parameterization of the road model

A circumference can be fully defined by three non-collinear points. Nevertheless, if we consider that the vehicle's trajectory is approximately tangent to the lane at the bottom of the image, then the center of the circumference is in $y = H$ (the origin of coordinates is set in the top-left corner of the image), and only two points are required: a point in the upper row $\mathbf{x}_u = (x_u, 0)$, and the point at the bottom row, $\mathbf{x}_b = (x_b, H)$, where $H$ is the height of the image. Figure 3 illustrates the proposed model.

We will define $\mathbf{x}_t = (x_{b,t}, x_{u,t}, w_t, \dot{x}_{b,t}, \dot{x}_{u,t}, \dot{w}_t)^\top$ as the state vector of the road model, where the subindex $t$ specifies the point in time. In Fig. 3 we can see that $x_b$ is the midpoint of the own lane at $y = H$, so that the position of the vehicle inside this lane is defined by the difference between $x_b$ and the default position $x_c$. In the case that the camera is installed centered in the vehicle, then $x_c$ is horizontally centered in the image. The midpoint of the lane at $y = 0$ is defined by $x_u$, and the width of the lane is defined by $w$. As it will be shown in next sections, the difference between $x_b$ and $x_u$ determines the curvature of the road. During the rest of the document, we will refer to two subvectors, $\mathbf{x}_{u,t} = (x_{u,t}, \dot{x}_{u,t})^\top$ and $\mathbf{x}_{b,t} = (x_{b,t}, w_t, \dot{x}_{b,t}, \dot{w}_t)^\top$. For convenience, we include the terms corresponding to the width of the lane in the subvector $\mathbf{x}_{b,t}$. In addition, although the width of the lanes typically is very constant, we have observed that in critical situations lanes may split and/or fuse for instance when exit ramps appear. In those situations, the variation of the width is non-constant, and the derivative term helps to adapt to such abrupt changes.

## 4 Bayesian inference method

The model adapts to the motion of the vehicle (lane changes) and to the variation of curvature of the road ahead exploiting concepts from the Bayesian inference theory. This probabilistic model provides a uniquely flexible and powerful framework in which we can easily combine information of very different nature within the observation and prior models, and which also allows to introduce interaction models, as described in [10, 14].

Bayesian inference methods provide estimations of the posterior density distribution $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ of $\mathbf{x}_t$, which is the parameterization of the road model, given all the observations up to current time, $\mathbf{z}_{1:t}$. The observations can be defined as the pixels in the image that most likely belong to lane markings. For the detection of these lane markings, we will use the detector proposed in [13] for its simplicity and effectiveness. An example of the result of this detector is shown in Fig. 2c.

In this section, we will first briefly introduce the concepts of Bayesian inference used for the design and implementation of the proposed particle filter.

### 4.1 Rao-Blackwellization

The analytical expression of the posterior density can be decomposed using the Bayes' rule as:

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) = kp(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{z}_{1:t-1}) \tag{1}$$

where $p(\mathbf{z}_t|\mathbf{x}_t)$ is the likelihood function that models how likely the measurement at time $t, \mathbf{z}_t$, would be observed given the system state vector, $\mathbf{x}_t$, and $p(\mathbf{x}_t|\mathbf{z}_{1:t-1})$ is the prediction information, since it provides all the information we know about the current state before the new observation is available. The constant $k$ is a scale factor that ensures that the density integrates to one.

The prediction distribution is given by integrating all the possible previous posterior distributions and with the dynamic model [1]:

$$p(\mathbf{x}_t|\mathbf{z}_{1:t-1}) = \int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})\,\mathrm{d}\mathbf{x}_{t-1} \tag{2}$$

Typically, this expression cannot be analytically solved. A popular tool to solve it is based on a recursive, sample-based approach, known as sequential Monte Carlo methods or particle filters. These methods approximate the posterior density function by a set of samples which are propagated in time and from which point-estimates of the target state-vector can be obtained [1, 10]. The more particles used, the better approximation, although when the dimensionality of the problem rises, the number of particles required to achieve a good representation of the posterior density function is increased as well. For the case of sequential importance sampling filters, the number of particles increases exponentially with the dimensionality of the problem [1].

We propose to apply the Rao-Blackwellization technique [9], which proceeds by assuming a sub-structure of the state-vector and marginalizing out (i.e. compute exactly) the part that can be considered linear. In our case, we intentionally divided the state vector $\mathbf{x}_t$ into a position and a curvature part, relating to $\mathbf{x}_{b,t}$, and $\mathbf{x}_{u,t}$, respectively. Therefore, the posterior can be defined as:

$$p(\mathbf{x}_t|\mathbf{z}_t) = p(\mathbf{x}_{b,t}, \mathbf{x}_{u,t}|\mathbf{z}_t) = p(\mathbf{x}_{u,t}|\mathbf{z}_t, \mathbf{x}_{b,t})p(\mathbf{x}_{b,t}|\mathbf{z}_t) \quad (3)$$

where we have declared explicitly that the only relevant measurement is $\mathbf{z}_t$ so we are assuming that the process is Markovian.

Regarding $p(\mathbf{x}_{b,t}|\mathbf{z}_t)$, we will consider that it represents that $\mathbf{x}_{b,t}$ depends linearly on $\mathbf{z}_t$ with an added Gaussian noise, and therefore can be solved analytically using the Kalman filter [1]. The Kalman filter represents the optimal solution, much more steady than the one provided by a particle filter, which can only reach the same accuracy increasing the number of particles. Note that we will save here many computational resources using a single Kalman filter that feeds all the samples of the particle filter. The first factor, $p(\mathbf{x}_{u,t}|\mathbf{z}_t, \mathbf{x}_{b,t})$, related to the curvature of the road, is considered as non-linear and thus we require the use of an approximation method. This expression states as well that $\mathbf{x}_{u,t}$ conditionally depends on $\mathbf{x}_{b,t}$.

## 5 Linear component

The linear component is composed of the transversal position of lane $x_{b,t}$ and its width $w_t$. We can obtain a complete observation of these magnitudes using the lane markings image (like the one in Fig. 2c). We propose to generate a pair of horizontal profiles from these images, one defined for the upper half $\mathbf{z}_{u,t}$ and the other for the bottom half $\mathbf{z}_{b,t}$. These profiles are computed as the sum of the intensity values of the pixels marked as lane markings at each column. We can see an example in Fig. 4. As shown, the accumulator of the lower half has two significant modes coincident with the left and right lane markings of the own lane, while, due to the curvature, these modes are more dilated and not so distinctive in $\mathbf{z}_{u,t}$.

For a given point in time $t$, we can compute the position of these modes by searching for one maximum in the surroundings of the previous estimations (these regions are shown in blue and green for the left and right lane markings respectively in Fig. 4). Should discontinuous or bad painted lane markings appear, we could find that one or both modes are not found. In that case, a prediction is used, until a predetermined number of frames without observations trigger a model reset.

The temporal coherence is provided by the use of the Kalman filter. Specifically, we use a constant-velocity model which models with accuracy the movement of the vehicle inside the lane, including the lane change movements. In this linear part of the algorithm, we work with $\mathbf{x}_{b,t}$ as the state vector whose state-equation is defined as:

$$\mathbf{x}_{b,t} = A\mathbf{x}_{b,t-1} + B\mathbf{u}_t + \mathbf{q}_t \quad (4)$$

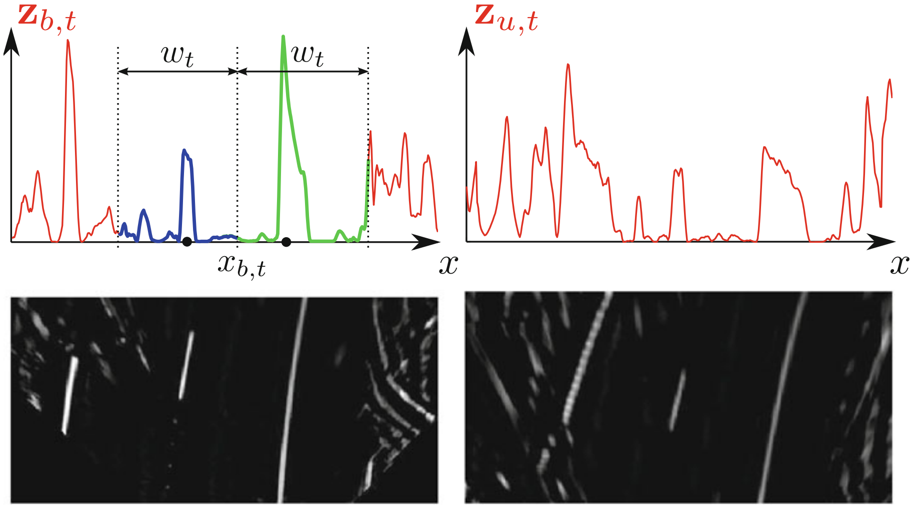


**Fig. 4** Profiles of the bottom (*left*) and upper (*right*) lane markings images

The transition matrix $A$ and the input control matrix $B$ are defined with a constant-velocity model as:

$$A = \begin{pmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{5}$$

and

$$B = \begin{pmatrix} 1 & 0 & 0 & 0 \end{pmatrix} \tag{6}$$

Note that this model only implies that the variation over a linear model is given by an uncertain component $\mathbf{q}_t$ with Gaussian profile. There is a linear relation between the state vector and the measurements $\mathbf{z}_t = H\mathbf{x}_{b,t} + \mathbf{r}_t$ given by the matrix:

$$H = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \tag{7}$$

Using this model, we can know the position of the vehicle inside its lane, and then we can predict situations in which the vehicle is departing from its lane when the estimated position is near to one of the lane markings. When the lane change occurs, it is required to shift the model position in the opposite direction of the movement. We utilize the user input $\mathbf{u}_t$ defined for the Kalman filter as the width of the lane, negative if the movement is to the left, $\mathbf{u}_t = -w_t$, and positive, $\mathbf{u}_t = w_t$ to the right. The use of such input control allows keeping the linearity and Gaussianity of the estimation procedure which is therefore not affected by the abrupt jump in the state-space.

## 6 Non-linear component

Let us now consider only the posterior density function of the non-linear part of the state-vector $p(\mathbf{x}_{u,t}|\mathbf{z}_t, \mathbf{x}_{b,t})$. We hypothesize that it can be approximated by a set of weighted samples:

$$p(\mathbf{x}_{u,t}|\mathbf{z}_t, \mathbf{x}_{b,t}) \approx \frac{1}{N_s}\sum_{i=1}^{N_s} \omega_t^{(i)} \delta(\mathbf{x}_{u,t} - \mathbf{x}_t^{(i)}) \tag{8}$$

As done in many works, the SIR particle filter [1]) is adopted. In this case, the weights can be computed as:

$$\omega_t^{(i)} \propto \omega_{t-1}^{(i)} p(\mathbf{z}_t, \mathbf{x}_{b,t}|\mathbf{x}_{u,t}^{(i)}) \tag{9}$$

Last equation means that the weights of the particles, or hypotheses of the state vector are proportional to the their likelihood. Since we have decreased the dimensionality of the problem to $\mathbf{x}_{u,t}$, the SIR particle filter behaves successfully [10]. Equation (9) can be rewritten if the likelihood function is decomposed using the multiplication rule:

$$\omega_t^{(i)} \propto \omega_{t-1}^{(i)} p(\mathbf{z}_t|\mathbf{x}_{b,t}, \mathbf{x}_{u,t}^{(i)}) \tag{10}$$

since $p(\mathbf{z}_t, \mathbf{x}_{b,t}|\mathbf{x}_{u,t}^{(i)}) = p(\mathbf{x}_{b,t}|\mathbf{x}_{u,t}^{(i)})p(\mathbf{z}_t|\mathbf{x}_{u,t}^{(i)}, \mathbf{x}_{b,t})$ and the first factor $p(\mathbf{x}_{b,t}|\mathbf{x}_{u,t}^{(i)}) = p(\mathbf{x}_{b,t})$ is a constant for all the samples.

### 6.1 Likelihood function

Let us consider an observation model that is the combination of two independent models, each one associated to one of the profiles:

$$p(\mathbf{z}_t|\mathbf{x}_{b,t}, \mathbf{x}_{u,t}^{(i)}) = p(\mathbf{z}_{u,t}|\mathbf{x}_{b,t}, \mathbf{x}_{u,t}^{(i)})p(\mathbf{z}_{b,t}|\mathbf{x}_{b,t}, \mathbf{x}_{u,t}^{(i)}) \tag{11}$$

The observation model for the upper profile is defined as follows:

$$p(\mathbf{z}_{u,t}|\mathbf{x}_{b,t}, \mathbf{x}_{u,t}^{(i)}) = \sum_{x=1}^{W} \mathbf{z}_{u,t}(x) f_u(x, \mathbf{x}_{b,t}, \mathbf{x}_{u,t}^{(i)}) \tag{12}$$

where $f_u(x, \mathbf{x}_{b,t}, \mathbf{x}_{u,t}^{(i)})$ is a synthetic profile composed of the information of the curves in $\{\mathbf{x}_{b,t}, \mathbf{x}_{u,t}^{(i)}\}$. The expression for $p(\mathbf{z}_{b,t}|\mathbf{x}_{b,t}, \mathbf{x}_{u,t}^{(i)})$ is completely analogous.

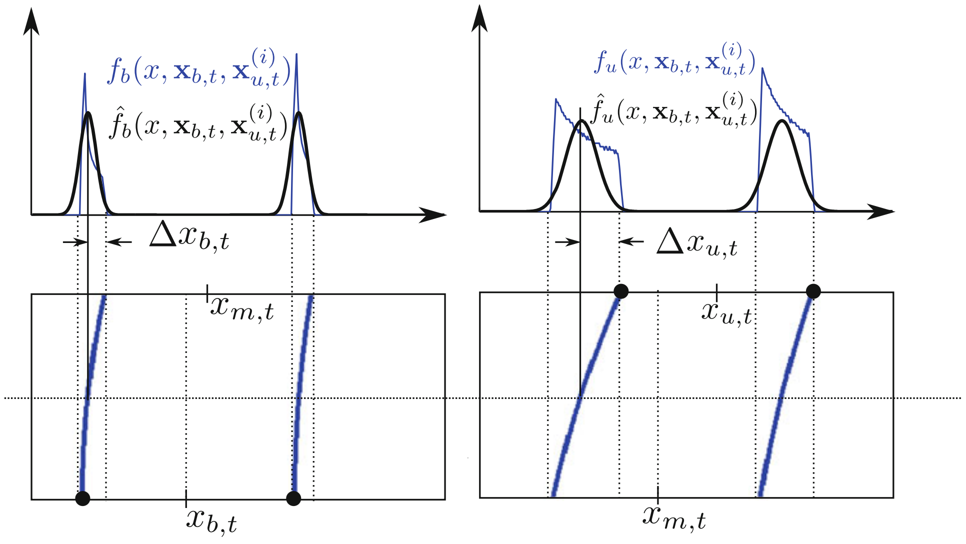

$f_b(x, \mathbf{x}_{b,t}, \mathbf{x}_{u,t}^{(i)})$
$\hat{f}_b(x, \mathbf{x}_{b,t}, \mathbf{x}_{u,t}^{(i)})$
$\Delta x_{b,t}$
$x_{m,t}$
$x_{b,t}$
$f_u(x, \mathbf{x}_{b,t}, \mathbf{x}_{u,t}^{(i)})$
$\hat{f}_u(x, \mathbf{x}_{b,t}, \mathbf{x}_{u,t}^{(i)})$
$\Delta x_{u,t}$
$x_{u,t}$
$x_{m,t}$

**Fig. 5** The ideal profile given by a curve is displayed with the Gaussian approximations, which are shifted by $\Delta x_{u,t}$ and $\Delta x_{b,t}$

The synthetic profile should be the profile of the curve obtained in ideal conditions: for instance, ideally, if a lane marking is a vertical line, its corresponding profile is a peak of minimum width. Nevertheless, due to the curvature, the profiles look like the examples $f_u(x, \mathbf{x}_{b,t}, \mathbf{x}_{u,t}^{(i)})$ and $f_b(x, \mathbf{x}_{b,t}, \mathbf{x}_{u,t}^{(i)})$ of Fig. 5.

The use of such $f_u(x, \mathbf{x}_{b,t}, \mathbf{x}_{u,t}^{(i)})$ and $f_b(x, \mathbf{x}_{b,t}, \mathbf{x}_{u,t}^{(i)})$ is subject to the computation of the profile of a corresponding synthetic image for the specific curve hypothesis. In practice, this is too much computationally intensive, and the approximations $\hat{f}_u(x, \mathbf{x}_{b,t}, \mathbf{x}_{u,t}^{(i)})$ and $\hat{f}_b(x, \mathbf{x}_{b,t}, \mathbf{x}_{u,t}^{(i)})$ are used.

Hence, $\hat{f}_u(x, \mathbf{x}_{b,t}, \mathbf{x}_{u,t}^{(i)})$ is defined as a mixture model:

$$\hat{f}_u(x, \mathbf{x}_{b,t}, \mathbf{x}_{u,t}^{(i)}) = \omega_u \rho + \sum_{c=1}^{L+1} \omega_c p_c(\mathbf{x}_{b,t}, \mathbf{x}_{u,t}^{(i)}) \tag{13}$$

composed of a uniform distribution, with value $\rho = \frac{1}{W}$, which is used to handle occlusions, and a mixture of functions $p_c(\mathbf{x}_{b,t}, \mathbf{x}_{u,t}^{(i)})$. The weighting factors $\omega$ ensure that the distribution integrates to one within the observation range. The uniform component ensures that even when a lane marking is non-existent (a discontinuous lane marking) or it is occluded by a vehicle or not measured for any other reason, the likelihood of a particle in the correct position will not be zero. In the absence of other information, all particles will receive the same likelihood and remain static (or following the estimated dynamics) until the occlusion ends.

The functions $p_c(\mathbf{x}_{b,t}, \mathbf{x}_{u,t}^{(i)})$ associated to each curve are defined as normal distributions centered at the expected positions of the lane markings (i.e. $w_t/2$ at left and right of the positions $x_{u,t}$ and $x_{b,t}$ respectively for the upper and bottom profiles, and shifted so that the gaussian profile covers the horizontal span of the circumference. This shift is computed as $\Delta x_{u,t} = (x_{u,t} - x_{m,t})/2$ and $\Delta x_{b,t} = -(x_{b,t} - x_{m,t})/2$ for curves to the left and with opposite signs for curves to the right. The position $x_{m,t}$ can be computed as $x_{m,t} = x_0 \pm \sqrt{r_0^2 - \left(\frac{H}{2}\right)^2}$, where $(x_0, y_0 = H)$ is the center of the corresponding circumference and $r_0^2$ its radius. These values can be obtained from three points as:

$$\begin{aligned} x_0 &= \frac{M_2}{2M_1} \\ r_0^2 &= x_0^2 + y_0^2 + \frac{M_4}{M_1} \end{aligned} \tag{14}$$

where $M_k$ are the coefficients of the expression of the circumference $(x^2 + y^2)M_1 - xM_2 + yM_3 - M_4 = 0$, that can be computed from:

$$\begin{vmatrix} x^2 + y^2 & x & y & 1 \\ x_{u,t}^2 & x_{u,t} & 0 & 1 \\ x_{b,t}^2 + H^2 & x_{b,t} & H & 1 \\ x_{u,t}^2 + (2H)^2 & x_{u,t} & 2H & 1 \end{vmatrix} = 0 \tag{15}$$

This expression also identifies the case in which the points are collinear when $M_1 = 0$.

As a result, given a particular hypothesis $\{\mathbf{x}_{b,t}, \mathbf{x}_{u,t}^{(i)}\}$, this observation model returns high likelihood values if the synthetic profile is similar to the observed profile, and low values if not.

## 7 Complexity reduction

The proposed method is specifically focused on achieving real-time performance. On the one hand, we have already shown the separation of the state vector into linear and non-linear parts. This way the Kalman filter can be applied for part of the state-vector which allows us to use the SIR particle filter in the subsequent non-linear stage for a reduced dimensionality. As defined in Sect. 3, the dimension of the complete state-vector is 6, while the dimensions explored with the particle filter are 2. Therefore, we are obtaining a $10^4$ reduction of the number of particles to use compared to a particle filter applied to the complete state vector (recall that the number of particles required by the filter increase exponentially with the dimension of the state [10].

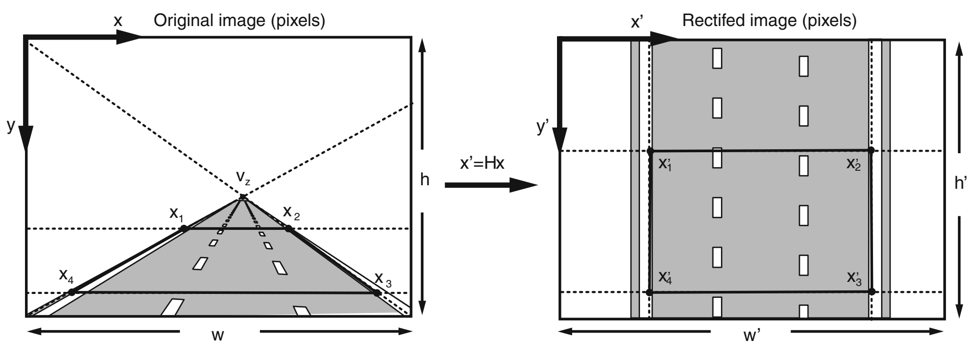


**Fig. 6** Inverse perspective mapping computed by means of 4 point correspondences

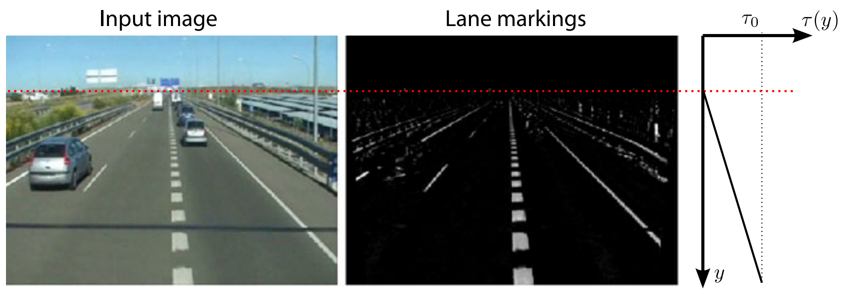


**Fig. 7** Lane markings detection using a step row filter [13] using a different expected width $\tau(y)$ according to the row of the image

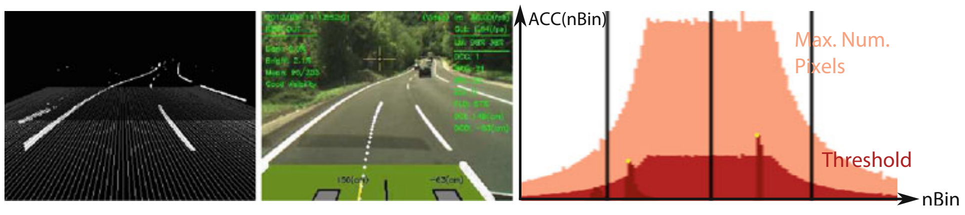


**Fig. 8** Example of projection of the regions of the image corresponding to each bin of the histogram. In the resulting histogram it can be seen that the right lane marking is the more significant in the image

**Table 1** Processing times of the two used computers applying in different run-view-record configurations

| Mode | Embedded PC | | | Desktop PC | | |
|---|---|---|---|---|---|---|
| | Fps | Total (ms) | Algorithm (ms) | Fps | Total (ms) | Algorithm (ms) |
| Run | 30.00 | 33.33 | 12.52 | 30.00 | 33.33 | 1.55 |
| Run + Rec | 29.60 | 35.00 | 12.66 | 30.00 | 33.33 | 1.58 |
| Run + Show | 27.49 | 38.00 | 13.41 | 30.00 | 33.33 | 1.61 |
| Run + Show + Rec | 25.18 | 41.00 | 13.54 | 30.00 | 33.33 | 1.65 |
| Run + 2Rec | 23.33 | 45.00 | 13.13 | 30.00 | 33.33 | 1.57 |
| Run + Show + 2Rec | 22.13 | 46.00 | 13.59 | 30.00 | 33.33 | 1.62 |

On the other hand, the implementation of the described method includes some modifications that significantly reduce the number of operations to be done. Initially, the proposed method uses an accumulator of lane markings pixels in the bird's-eye view image (see Sect. 5). The generation of such image can be done by applying the homography $H$ that links the image plane and the road plane for all the pixels of the bird's-eye view, searching for the pixel in the original domain from which take its intensity value: $\mathbf{x} = H^{-1}\mathbf{x}'$. If the homography is constant (i.e. the perspective of the scene can be assumed constant), it is convenient to precompute all the possible transformations and store them in memory as a 2D–2D map, so that the transformations are substituted by accesses to memory in runtime.

Nevertheless, we propose an even faster approach that does not generate the bird's-eye view but allows us to obtain the target accumulation profiles directly. We can generate a 2D–1D look-up table (LUT) that links points in the original image to columns in the bird's-eye view. More generally, we can define these columns as bins of the accumulators, such that we directly map from the original 2D image to the 1D accumulators. This way we have substituted the need of a memory map of $W \times H$ integer values by a $W \times 1$ sized map. Compared to the generation of the image, we avoid the computation of $W \times H$ linear projetions and writing them into memory.

The LUT can be computed as follows. Once the vanishing point has been computed, we can obtain the homography $H$ using 4 point correspondences (as illustrated in Fig. 6). We can now project each pixel position of the original image as $\mathbf{x} = H\mathbf{x}$, obtain the corresponding bin $b$ and store it into the LUT. This way, the LUT is defined as:

$$L(x, y) = b. \tag{16}$$

The detection of the lane markings is modified such that it is now applied on the original image, and not in the (non-existing) bird's-eye view. For that purpose, we slightly modify the proposed detector [13] to search for intensity bumps in the image with a width defined by a function of the $y$ coordinate. If we define that the expected lane marking width at the bottom of the image is $\tau_0$, we can set a linear decreasing function according to the row of the

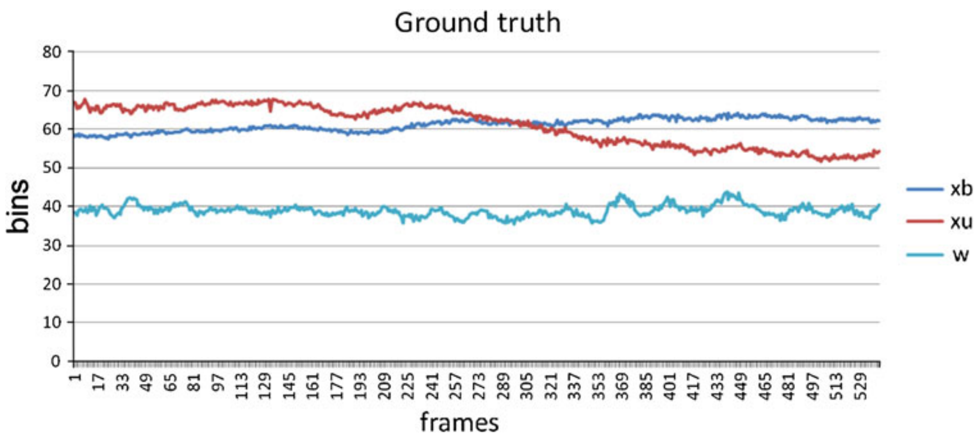


**Fig. 9** Ground truth of a example sequence. The graph shows a curve to the right until frame 275 ($x_u > x_b$), then a straight path during 50 frames ($x_u \simeq x_b$), and then a curve to the left ($x_u < x_b$)

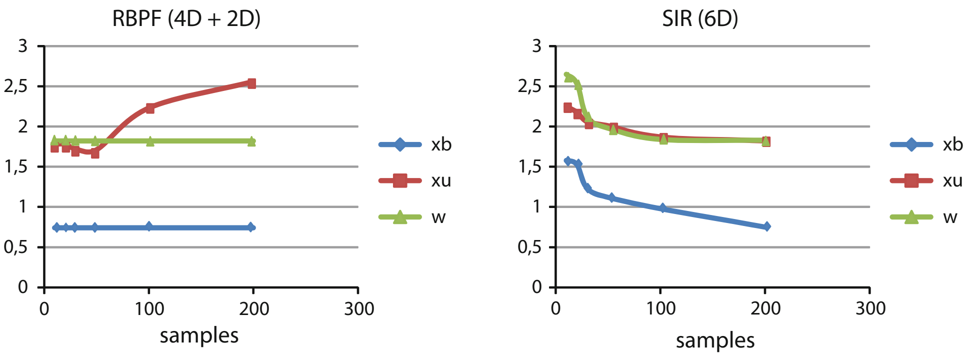


**Fig. 10** Mean absolute error (in bins) for the proposed algorithm (*RBPF*) and the sequential importance sampling method (*SIR*) for different number of samples

image in which the filter is applied so that at the height of the vanishing point, the expected width is zero. Figure 7 shows an example of this detector.

For each pixel detected as belonging to a lane marking, we increase the bin of the accumulator defined by the LUT with the intensity value of the pixel:

$$\mathbf{z}_{b,t}(L(x,y)) = \mathbf{z}_{b,t}(L(x,y)) + I(x,y) \tag{17}$$

for pixels corresponding to the bottom half of the bird's-eye view domain, and an analougous expression for the accumulator of the upper half $\mathbf{z}_{u,t}$.

This operation implies one access to memory and one sum, and is only applied to the pixels detected by the lane markings detector, which typically represent no more than 2–5 % of the pixels of the whole image.

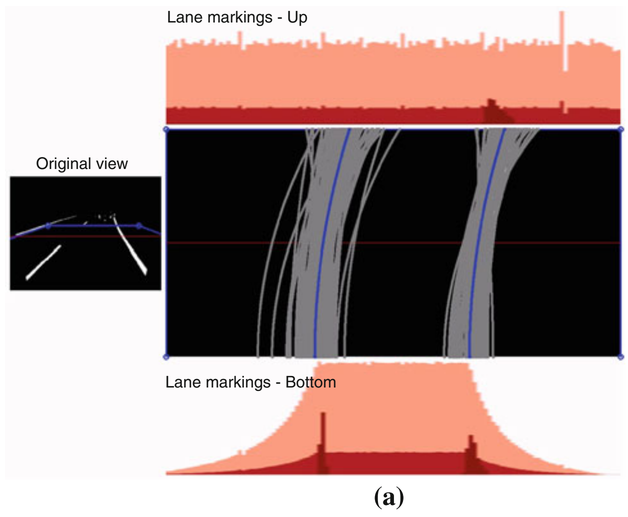
Figure 8 illustrates the computation of the accumulator. The detected lane markings are shown in white in the left-most image with the superimposed perspective bins (light gray: near histogram corresponding to the bottom half of the bird's-eye view, dark gray: far histogram corresponding to the upper half of the bird's-eye view). The example accumulator is shown in the right-most figure. In that figure we can observe two clear peaks at left and right of the center, corresponding to the two corresponding observed lane markings of the image. Due to the perspective effect and to the boundaries of the images, not all bins can count the same number of pixels. As shown, the central bins cover a larger number of pixels, while the lateral ones have a decreasing number of pixels the greater their distance to the center (this number is illustrated in the right-most figure as Max. Num. Pixels). Therefore, the detection of local maxima has to be done using an adaptive threshold, which is set as a percentage of the maximum number of possible pixels per bin. We have used a variable threshold that adapts itself to the illumination conditions (35 % works fine for sunny and well illuminated sequences, while 20 % should be used in dark scenarios such as tunnels or night). As shown in the right-most figure of Fig. 8, the threshold is a scaled function of the number of pixels per bin.

## 8 Tests and discussion

In this section, we face both the evaluation of the system functionalities and its performance in terms of real-time operation.

### 8.1 Implementation details

The proposed method has been implemented in C++ as a multi-platform software, and built and tested in two



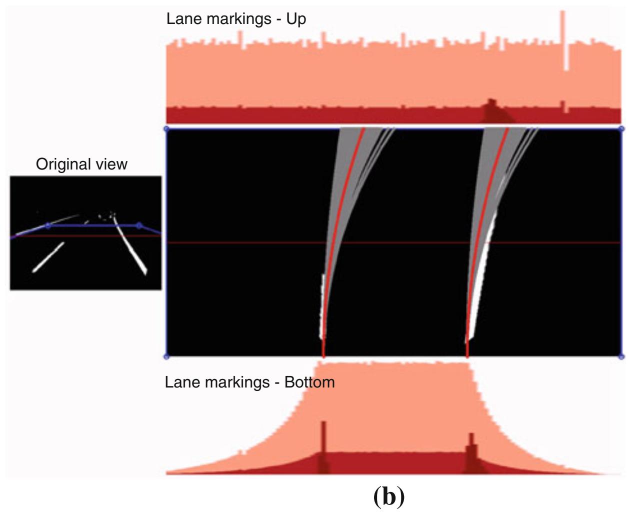


**Fig. 11** Intuitively, this figure illustrates that the SIR particle filter (**a**) needs to span a larger domain than the proposed approach (**b**), which is much more directed to the estimation of the curvature: in *gray* we can see each sample of the particle filters, which hypothesize a certain curvature, and in *blue* and *red* we can see the point-wise estimates of the SIR and proposed particle filter, respectively

different computers to compare its performance in general purpose CPUs and embedded systems. On the one hand, for the design and debugging stages of the project, we have used a Core 2 Quad CPU Q8300 @ 2.5 GHz, with 4 GB RAM, using Windows 7. On the other hand, for the installation into a real vehicle, we have used an embedded computer with Intel Atom CPU N270 @ 1.6 GHz, 1 GB RAM with Ubuntu SERVER 11.10 installed.

During the evaluation stage, it was necessary to have a visual output of the system, so we included a visualization module that allowed us to check the behavior of the method. In addition, we added a video capture module for both the input video stream (to store original videos from the camera) and the output video with the superimposed road model (to store the result of the system). All these modules can be switched on and off with a configuration file, so that we can evaluate the system according to the different enabled modules. In that sense we can study the cost of having processing, rendering and storing capabilities at the different CPUs.

Table 1 shows the processing times of the two considered CPUs for the different mentioned configurations. The embedded processor spends about 12 ms running the proposed algorithm (denoted in the table as Algorithm (ms)), while the desktop PC spends 1.55 ms. The total (ms) time is the sum of the time spent in processing the algorithm, renderisation and recording tasks, and waiting for the next frame to appear. Therefore, the minimum value is the elapsed time between frames (33.33 ms for 30 fps).

For the desktop PC, the total time never exceeds 33.33 ms. The embedded PC runs at 30 fps if renderisation and recording are switched off. In the embedded CPU, the visualization of the video output slows the system down to 27.49 fps, since it makes use of CPU resources to show the graphic window (the embedded computer did not have a direct connection to a monitor, but an Ethernet connection was established with a host computer which received the visualization information through SSH connection). The video storage implied a similar slow down in the embedded computer, getting 29.60 fps recording one video stream, and 23.33 fps recording two.

We have evaluated the impact of using more or less samples in the proposed particle filter, and also compared its accuracy with a particle filter that spans all the dimensions of the state vector (using the well-known sequential importance resampling algorithm). For that purpose, we have generated the ground truth position of the lane, its width and curvature (as parameterized in our road model) of a road in a sequence of 500 frames, which are used to illustrate the performance of the system (it contains a curve to right, a straight stretch and then a curve to left; the ground truth values can be seen in Fig. 9).

We have run the proposed algorithm and compared the values of $x_{b,t}$, $x_{u,t}$, and $w_t$ with those of the ground truth. The mean absolute error metric is used, and the obtained values are shown in Fig. 10.

From the figure, we can observe that the errors of $x_{b,t}$ and $w_t$ are constant for RBPF, since we are using the linear Kalman filter and thus they are not affected by a variation of the number of samples of the particle filter. The error of $x_{u,t}$ finds a minimum near 50 samples, which is coherent with our hypothesis that a low number of samples is enough to sample that reduced dimension space. The increase of the error with the number of particles is possibly due to the excessive spread of particles that reach local maxima in the likelihood function and thus survive to the resampling and increase the average error. In the case

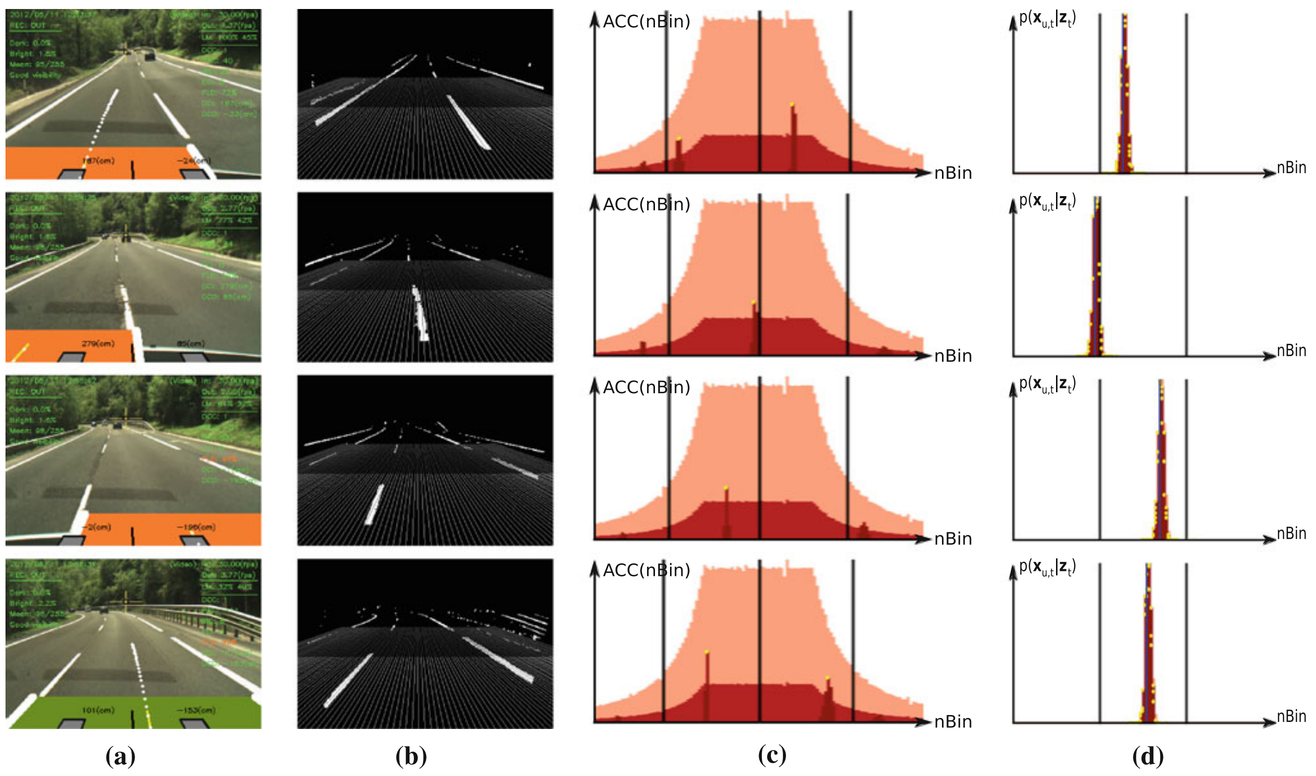


**Fig. 12** Example sequence of a lane change with curvature. In **a** we can see the change of color in which the system paints the road according to the situation: *green* for normal driving, *orange* in lane departure warning status

**Table 2** Detection of lane changes

| | Duration | TP | FN | FP | R | P |
|---|---|---|---|---|---|---|
| Sunny1 | 25′35″ | 14 | 1 | 0 | 0.933 | 1.000 |
| Sunny2 | 24′39″ | 21 | 1 | 0 | 0.954 | 1.000 |
| Tunnels1 | 30′51″ | 5 | 1 | 0 | 0.833 | 1.000 |
| Tunnels2 | 22′51″ | 23 | 2 | 1 | 0.920 | 0.958 |
| Rainy1 | 16′00″ | 13 | 2 | 1 | 0.867 | 0.928 |
| Rainy2 | 9′27″ | 13 | 0 | 1 | 1.000 | 0.928 |
| Total | 129′23″ | 89 | 7 | 3 | 0.927 | 0.967 |

*TP* true positives, *FP* false positives, *FN* false negatives, *R* recall = TP/(TP + FN), *P* precision = TP/(TP + FP)

of the SIR algorithm, we have found that using a low number of particles tends to give incorrect results, with high error (above 2 bins) since 10–50 samples are not enough to densely sample a six-dimension vector. Therefore, the higher the number of samples, the better the accuracy. We stopped computing results at 200 samples due to the excessive computational complexity.

We have illustrated this behavior in Fig. 11 in which the two algorithms seek for the best fit of the circumference model in the bird's-eye view. The SIR particle filter tends to spread particles according to the normal system noise in the six dimensions comprised in the state vector. In contrast, our algorithm, using the location and width of the lane at the bottom profile given by the linear part, focuses much better the determination of the curvature by sampling only in the two remaining dimensions in $\mathbf{x}_{u,t}$.

## 8.2 System evaluation

We have evaluated the performance of the system in a set of sequences with a total length of 129 min, in roads and highways, with varying illumination conditions (direct sun light, cloudy days, rain, night, etc). One way to objectively determine the correct functioning of the system is to check the detection of lane changes. The detection of such events shows that the system is well detecting the position of the lane markings. Lane changes are challenging situations that usually involve fast motion and partial absence of measurements (typically, only one lane marking is observed during such manoeuvre). Therefore, we can assume that the system is able to monitorize when the vehicle is getting out of its own lane if lane changes are detected correctly.

Figure 12 shows an example sequence featuring a lane change, which has been correctly detected by the system. For a better understanding, the image of detected lane marking pixels is shown in Fig. 12b. The position of the vehicle is depicted showing the position of its frontal wheels and the relative distance from the wheels to the closest lane marking. We can see in the figure that as the

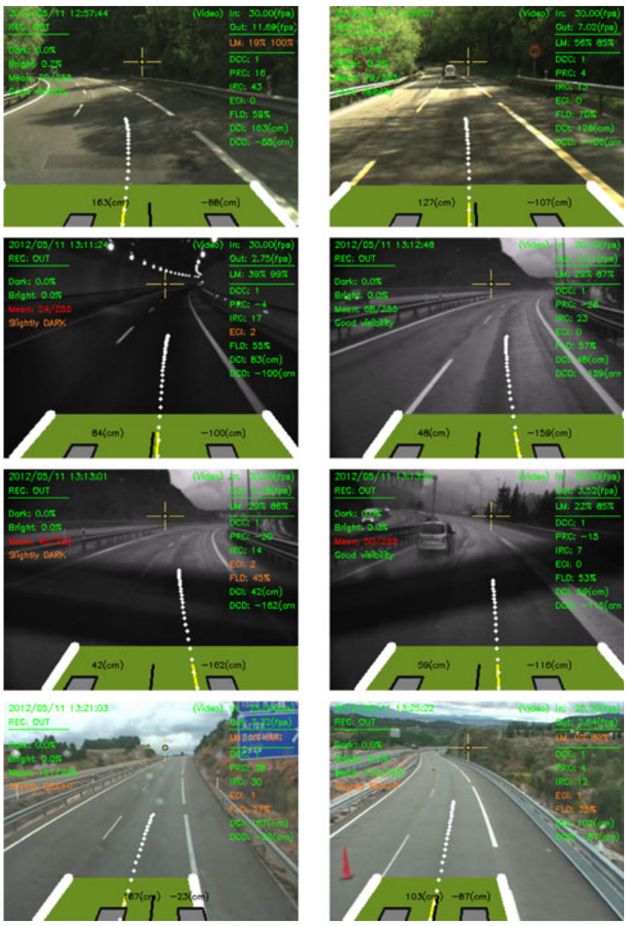


Fig. 13 Examples of the visualization the computed road model in different scenarios, including curves, tunnels, rain drops, presence of wipers, shadows, vehicles, etc

vehicle moves right, the system detects that the distance between the right wheel and the lane marking decreases. In that moment the system emits a signal to the driver (through the HMI). The lane change is completed when the central position of the vehicle exceeds the lane marking, and then the lane model switches to the new own lane.

In Fig. 12c, d, we can see the profiles of the perspective accumulator and the distribution of the samples of the particle filter, respectively. We can see in this example that the search for maxima is complex in these situations, since there could be absence of observations above the threshold due to the presence of discontinuous lane markings (first row). In the third row, we can see that when we get the very next detection, the lane marking is not at the right side anymore, but in the left. As stated, whenever there are lane changes, we apply the input control vector $\mathbf{u}_t$ to the Kalman filter.

Table 2 shows the statistics of lane change detections. The recall and precision values are computed to show the performance of the proposed method. The recall, which is related to the number of correctly detected events, give values above 92 %, while the precision, related to the quality of the detections, is above 96 %.

The accurate estimation of the curvature in this environment is pretty difficult, and heavily depends on the correct estimation of the lane position (provided by the linear component of our method), and the visibility of the curvature ahead. Indeed, the curvature can only be computed when the lanes exist, are visible at a far distance, and the position of the lane has been correctly detected. Our method switches off the computation of the curvature

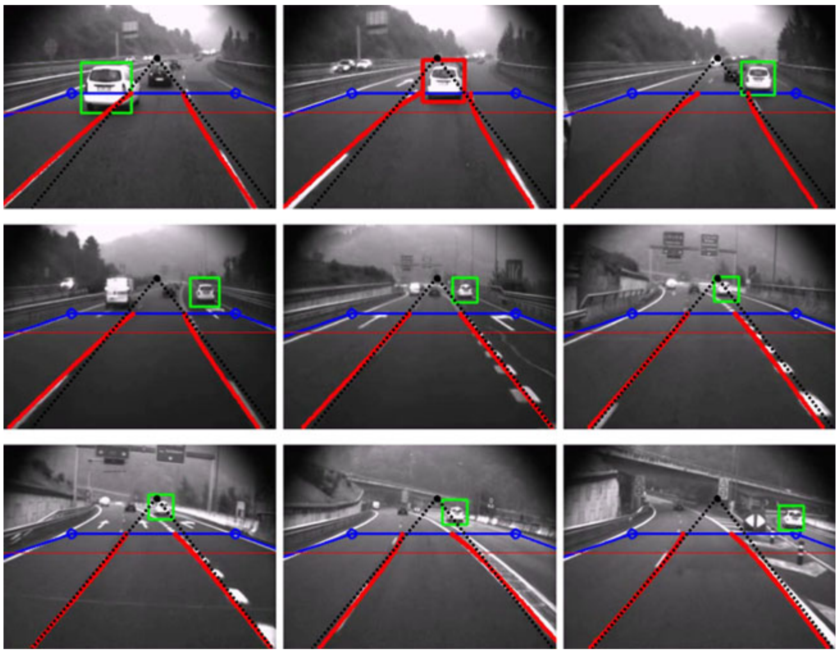


Fig. 14 Detecting the curvature of the road may help to reduce false alarms of vehicle detections methods: (from *left to right* and *top to bottom*) a vehicle is entering our lane; the vehicle move to another lane; the curvature allow to know that the vehicle is in the adjacent lane

in the situations in which the reliability of the lane markings detection does not exceed a defined threshold. In general, for the sequences we have analyzed, the curvature is accurate in most situations in which there is at least one continuous lane markings, including different illumination conditions. Some key frames in Fig. 13 illustrate the behavior of the proposed method in different scenarios.

As a special interesting case, we illustrate in Fig. 14 the potential use of our method to reduce the number of false alarms of obstacle detection or safety distance systems. As mentioned previously, sensors like radar or laser are not aware of the curvature of the road, and thus trigger false alarms when vehicles or obstacles appear in front of the own vehicle even though they are not in the own lane (and trajectory).

## 9 Future work

Our intention is to adapt the proposed framework to include other ADAS services such as obstacle detection, traffic sign recognition, etc. We are confident the structure of our platform can hold with other computer vision algorithms running in parallel. The frame rates achieved for the lane modeling, including curvature, make us think that additional methods with moderate complexity can be placed all together and keep real-time performance. Particularly, we are currently designing multiple vehicle tracking, traffic sign detection and driver attention monitorization methods.

## 10 Conclusions

In this paper, we have introduced an advanced driver assistance system that contains a method for lane departure warning and the estimation of the curvature of the road, using a monocular sensor connected to an embedded PC in which the results can be materialized as videos, monitorized in screens, or sent to remote hosts through shared memory capabilities

The proposed method works in real-time for cameras capturing at 30 Hz in the embedded PC. The algorithm runs in less than 2 ms for standard PCs. At the heart of our contribution, we apply the concepts of the Rao-Blacwellization to split the processing into two stages. On the one hand, the linear part determines the transversal position of the vehicle at its own lane. On the other hand, the nonlinear part generates hypotheses of the curvature, evaluates and combines them using a particle filter.

The obtained results make us confident that our system can be integrated in real vehicles, providing driver assistance services in a wide range of outdoor scenarios, including situations of sudden illumination changes, night, absence of measurements, presence of other vehicles, etc.

**Acknowledgments** This work has been partially supported by the program ETORGAI 2011-2013 of the Basque Government under project IEB11. This work has been possible thanks to the cooperation with Datik–Irizar Group for their support in the installation, integration and testing stages of the project.

## References

1. Arulampalam, M.S., Maskell, S., Gordon, N., Clapp, T.: A tutorial on particle filters for online nonlinear/nonGaussian Bayesian tracking. IEEE. Trans. Signal. Process. **50**(2), 174–188 (2002)
2. Borkar, A., Hayes, M., Smith, M.T.: Lane detection and tracking using a layered approach. In: Proceedings of Advanced Concepts for Intelligent Vision Systems, pp. 474–484. (2009)
3. Borkar, A., Hayes, M., Smith, M.T.: A new multi-camera approach for lane departure warning. In: Proceedings of Advanced Concepts for Intelligent Vision Systems, pp. 59–69. (2011)
4. Chiu, C.-C., Chung, M.-L., Chen, W.-C.: Real-time front vehicle detection algorithm for an asynchronous binocular system. J. Inf. Sci. Eng. **26**(3), 735–752 (2010)
5. Corridori, C., Zanin, M.: High curvature two-clothoid road model estimation. In: IEEE Proceedings of Intelligent Transportation Systems Conference, pp. 630–636. (2004)
6. Danescu, R., Nedevschi, S., Meinecke, M.-M., To, T.-B.: A stereovision-based probabilistic lane tracker for difficult road scenarios. In: IEEE Proceedings of Intelligent Vehicles Symposium, pp. 536–541. (2008)
7. del Blanco, C.R., Jaureguizar, F., García, N.: An advanced Bayesian model for the visual tracking of multiple interacting objects. EURASIP J. Adv. Signal Process. **130** (2011)
8. Dickmanns, E.D., Graefe, V.: Dynamic monocular machine vision. MachineVision Appl. **1**(4), 223–240 (1988)
9. Doucet, A., Gordon, N.J., Krishnamurthy, V.: Particle filters for state estimation of jump Markov linear systems. IEEE Trans. Signal Process. **49**(3), 613–624 (2001)
10. Khan, Z., Balch, T., Dellaert, F.: MCMC-based particle filtering for tracking a variable number of interacting targets. IEEE Trans. Pattern Anal. Mach. Intell. **27**(11), 1805–1819 (2005)
11. Kim, Z.: Robust lane detection and tracking in challenging scenarios. IEEE. Trans. Intell. Transp. Syst. **9**(1), 16–26 (2008)
12. McCall, J.C., Trivedi, M.M.: Video-based lane estimation and tracking for driver assistance: survey, system and evaluation. IEEE. Trans. Intell. Transp. Syst. **7**(1), 20–37 (2006)
13. Nieto, M., Arróspide, J., Salgado, L.: Road environment modeling using robust perspective analysis and recursive Bayesian segmentation. Mach. Vis. Appl. **22**(6), 927–945 (2011)
14. Smal, I., Niessen, W., Meijering, E.: Advanced particle filtering for multiple object tracking in dynamic fluorescence microscopy images. In: IEEE Proceedings of International Symposium on Biomedical Imaging, pp. 1048–1051. (2007)
15. Southall, B., Taylor, C.J.: Stochastic road shape estimation. In: Proceedings of International Conference on Computer Vision, vol. 1, pp. 205–212. (2001)
16. Wang, Y., Teoh, E.K., Shen, D.: Lane detection and tracking using B-snakes. Image. Vis. Comput. **22**, 269–289 (2004)

## Author Biographies

**Marcos Nieto** received the Ingeniero de Telecomunicación degree in 2005 and the Ph.D. degree in 2010 both from the E.T.S.Ing. Telecomunicación (ETSIT) of the Universidad Politécnica de Madrid

(UPM), Spain. He currently works in the Vicomtech-ik4 research center, in which he is working in the area of Intelligent Transportation Systems and Engineering. His actual research interests include the use of optimization methods for probabilistic models in computer vision as well as the design of H.264/AVC video codecs.

**Andoni Cortés** received the M.S. degree in computer science from the University of the Basque Country, Donostia–San Sebastian, Spain, in 2001. He currently works as a researcher of the ITS and Engineering Area of Vicomtech, Donostia–San Sebastian, Spain. His research interests include pattern recognition and augmented reality.

**Oihana Otaegui** received her M.S. and Ph.D. degrees in mechanical engineering from the Tecnun, University of Navarra, Donostia–San Sebastian, Spain in 1999 and 2005 respectively. She is currently the head of the ITS and Engineering Area of Vicomtech, Donostia–San Sebastian, Spain. Her research interests include satellite navigation and transport fields.

**Jon Arróspide Laborda** received the Ingeniero de Telecomunicación degree in 2006 from the TECNUN School of the Universidad de Navarra, Spain. Since 2007 he is a member of the Grupo de Tratamiento de Imágenes (GTI, Image Processing Group) of the Universidad Politécnica de Madrid (UPM), where he is currently working as a Ph. D. student. His research focuses on object detection and tracking, especially related to intelligent transport systems.

**Luis Salgado** received the Ingeniero de Telecomunicación degree in 1990 and the Ph.D. degree in 1998, both from the E.T.S. Ingenieros de Telecomunicación, Universidad Politécnica de Madrid (UPM), Spain. Since 1990, he has been a member of the Grupo de Tratamiento de Imágenes (GTI, Image Processing Group) of the UPM where he currently work as an Associate Professor. He is Associate Editor of the Journal of Real-Time Image Processing, and has been member of the Scientific and Program Committees of several international conferences and has been Auditor and Evaluator of European research programs since 2002. He has participated in many national and international research projects, and his professional interests include video analysis, processing and coding.