Noname manuscript No.
(will be inserted by the editor)

Vehicle tracking and classification in challenging scenarios via slice sampling

Marcos Nieto*1, Luis Unzueta[1], Javier Barandiaran[1], Andoni Cortés[1], Oihana Otaegui[1] and Pedro Sánchez[2]

[1]Vicomtech-ik4, Mikeletegi Pasealekua 57, Donostia-San Sebastián 20009, Spain

[2]IKUSI, Miketelegi Pasealekua 180, Donostia-San Sebastián 20009, Spain

***Corresponding author: mnieto@vicomtech.org**

the date of receipt and acceptance should be inserted later

Abstract This article introduces a 3D vehicle tracking system in a traffic surveillance environment devised for shadow tolling applications. It has been specially designed to operate in real time with high correct detection and classification rates. The system is capable of providing accurate and robust results in challenging road scenarios, with rain, traffic jams, casted shadows in sunny days at sunrise and sunset times, etc. A Bayesian inference method has been designed to generate estimates of multiple variable objects entering and exiting the scene. This framework

Address(es) of author(s) should be given

allows easily mixing different nature information, gathering in a single step observation models, calibration, motion priors and interaction models. The inference of results is carried out with a novel optimization procedure that generates estimates of the maxima of the posterior distribution combining concepts from Gibbs and slice sampling. Experimental tests have shown excellent results for traffic-flow video surveillance applications that can be used to classify vehicles according to their length, width, and height. Therefore, this vision-based system can be seen as a good substitute to existing inductive loop detectors.

Keywords: vehicle tracking; Bayesian inference; MRF; particle filter; shadow tolling; ILD; slice sampling; real time.

1 Introduction

The advancements of the technology as well as the reduction of costs of processing and communications equipment are promoting the use of novel counting systems by road operators. A key target is to allow free flow tolling services or shadow tolling to reduce traffic congestion on toll roads.

This type of systems must meet a set of requirements for its implementation. Namely, on the one hand, they must operate real time, i.e. they must acquire the information (through its corresponding sensing platform), process it, and send it to a control center in time to acquire, process, and submit new events. On the other hand, these systems must have a high reliability in all situations (day, night, adverse weather conditions). Finally, if we focus on shadow tolling systems, then the system is considered to be working if it is not only capable of counting vehicles, but also classifying them according to their dimensions or weight.

There are several existing technologies capable of addressing some of these requirements, such as intrusive systems like radar and laser, sonar volumetric estimation, or counting and mass measurement by inductive loop detectors (ILDs). The latter, being the most mature technology, has been used extensively, providing good detection and classification results. However, ILDs present three significant drawbacks: (i) these systems involve the excavation of the road to place the sensing devices, which is an expensive task, and requires disabling the lanes in which the ILDs are going to operate; (ii) typically, an ILD sensor is installed per lane, so that there are miss-detections and/or false positives when vehicles travel between lanes; and (iii) ILD cannot correctly manage the count in situations of traffic congestion, e.g. this technology cannot distinguish two small vehicles circulating slowly or standing over an ILD sensor from a large vehicle.

Technologies based on time-of-flight sensors represent an alternative to ILD, since they can be installed with a much lower cost, and can deliver similar counting and classifying results. There are, however, as well, two main aspects that make operators reluctance to use them: (i) on the one hand, despite the existence of the technology for decades, applied for counting and classification in traffic surveillance is relatively new, and there are no solutions that represent real competition against ILD in terms of count and classification results; and (ii) these systems can be called intrusive with the electromagnetic spectrum because they emit a certain amount of radiation that is reflected on objects and returns to the sensor. The emission of radiation is a contentious point, since it requires to meet the local regulations in force, as well as to overcome the reluctance of public opinion regarding radiation emission.

Recently, a new trend is emerging based on the use of video processing. The use of vision systems is becoming an alternative to the mentioned technologies. Their main advantage, as well as radar and laser systems one, is that their cost is much lower than ILDs, while its ability to count and classify is potentially the same. Moreover, as it only implies image processing, no radiation is emitted to the road, so they can be considered completely non-intrusive. Nevertheless, vision-based systems should still be considered as in a prototype stage until they are able to achieve correct detection and classification rates high enough for real implementation in free tolling or shadow tolling systems. In this article, a new vision-based system is introduced, which represents a real alternative to traditional intrusive sensing systems for shadow tolling applications, since it provides the required levels of accuracy and robustness to the detection and classification tasks. It uses a single camera and a processor that captures images and processes them to generate estimates of the vehicles circulating on a road stretch.

As a summary, the proposed method is based on a Bayesian inference theory, which provides an unbeatable framework to combine different nature information. Hence, the method is able to track a variable number of vehicles and classify them according to their estimated dimensions. The proposed solution has been tested with a set of long video sequences, captured under different illumination conditions, traffic load, adverse weather conditions, etc., where it has been proven to yield excellent results.

2 Related work

Typically, the literature associated with traffic video surveillance is focused on counting vehicles using basic image processing techniques to obtain statistics about lane usage. Nevertheless, there are many works that aim to provide more complex estimates of vehicle dynamics and dimensions to classify them as light or heavy. In urban scenarios, typically at intersections, the relative rotation of the vehicles is also of interest [1].

Among the difficulties that these methods face, shadows casted by vehicles are the hardest one to tackle robustly. Perceptually, shadows are moving objects that differ from the background. This is a relatively critical problem for single-camera setups. There are many works that do not pay special attention to this issue, which dramatically limits the impact of the proposed solutions in real situations [2–4].

Regarding the camera view point, it is quite typical to face the problem of tracking and counting vehicles with a camera that is looking down on the road from a pole, with a high angle [5]. In this situation, the problem is simplified since the perspective effect is less pronounced and vehicle dimensions do not vary significantly and the problem of occlusion can be safely ignored. Nevertheless, real solutions shall consider as well the case of low angle views of the road, since it is not always possible to install the camera so high. Indeed, this issue has not been explicitly tackled by many researchers, being of particular relevance the work by [3], which is based on a feature tracking strategy.

There are many methods that claim to track vehicles for a traffic counting solution but without explicitly using a model whose dimensions or dynamics are fitted to the observations. In these works, the vehicle is simply treated as a set of foreground pixels [4], or as a set of feature points [2,3].

Works more focused on the tracking stage, typically define a 3D model of the vehicles, which are somehow parameterized and fitted using optimization procedures. For instance, in [1], a detailed wireframe vehicle model that is fitted to the observations is proposed. Improvements on this line [6,7] comprise a variety of vehicle models, including detailed wireframe corresponding to trucks, cars, and other vehicle types, which provide accurate representations of the shape, volume, and orientation of vehicles. An intermediate approach is based on the definition of a cuboid model of variable size [8,9].

Regarding the tracking method, some works have just used simple data association between detections in different time instants [2]. Nevertheless, it is much more efficient and robust to use Bayesian approaches like the Kalman filter [10], the extended Kalman filter [11], and, as a generalization, particle filter methods [8,12]. The work by [8] is particularly significant in this field, since they are able to efficiently handle entering and exiting vehicles in a single filter, being as well able to track multiple objects in real time. For that purpose, they use an MCMC-based particle filter. This type of filter has been widely used since it was proven to yield stable and reliable results for multiple object tracking [13]. One of the main advantages of this type of filters is that the required number of particles is a linear function of the number of objects, in contrast to the exponentially growing demand of traditional particle filters (like the sequential importance resampling algorithm [14]).

As described by [13], the MCMC-based particle filter uses the Metropolis–Hastings algorithm to directly sample from the joint posterior distribution of the complete state vector (containing the information of the objects of the scene). Nevertheless, as happens with many other sampling strategies, the use of this algorithm guarantees the convergence only when using an infinite number of samples. In real conditions, the number of particles shall be determined experimentally. In traffic-flow surveillance applications, the scene will typically contain from none to 4 or 5 vehicles, and the required number of particles should be around 1,000 (the need of as few as 200 particles was reported in [8]).

In the authors opinion, this load is still excessive, and thus have motivated the proposal of a novel sampling procedure devised as a combination of the Gibbs and Slice sampling [15]. This method is more adapted to the scene proposing moves on those dimensions that require more change between consecutive time instants. As it will be shown in next sections, this approach requires an average between 10 and 70 samples to provide accurate estimates of several objects in the scene.

Besides, and as a general criticism, almost all of the above-mentioned works have not been tested with large enough datasets to provide realistic evaluations of its performance. For that purpose, we have focused on providing a large set of tests that demonstrate how the proposed system works in many different situations.

3 System overview

The steps of the proposed method are depicted in Fig. 1, which shows a block diagram and example images of several intermediate steps of the processing chain. As shown, the first module corrects the radial distortion of the images and applies a plane-to-plane homography that generates a bird's-eye view of the road. Although the shape of the vehicles appear in this image distorted by the perspective, their speed and position are not, so that this domain helps to simplify prior models and the computation of distances.

The first processing step extracts the background of the scene, and thus generates a segmentation of the moving objects. This procedure is based on the well-known codewords approach, which generates an updated background model through time according to the observations [16].

The foreground image is used to generate blobs or groups of connected pixels, which are described by their bounding boxes (shown in Fig. 1 as red rectangles). At this point, the rest of the processing is carried out only on the data structures that describe these bounding boxes, so that no other image processing stage is required. Therefore, the computational cost of the following steps is significantly reduced.

As the core of the system, the Bayesian inference step takes as input the detected boxes, and generates estimates of the position and dimensions of the vehicles in the scene. As it will be described in next sections, this module is a recursive scheme that takes into account previous estimates and current observations to generate accurate and coherent results. The appearance and disappearance of objects is controlled by an external module, since, in this type of scenes, vehicles are assumed to appear and disappear in pre-defined regions of the scene.

4 Camera calibration

The system has been designed to work, potentially, with any point of view of the road. Nevertheless, some perspectives are preferable, since the distortion of the projection on the rectified view is less pronounced. Figure 2 illustrates the distortion effect obtained with different views of the same road. As shown, to reduce the perspective distortion, it is better to work with sequences captured with cameras installed at more height over the road, although this is not always possible, so that the system must cope also with these challenging situations.

In any case, the perspective of the input images must be described, and it can be done obtaining the calibration of the camera. Although there are methods that can retrieve the rectified views of the road without knowing the camera calibration [5], we require it for the tracking stage. Hence, we have used a simple method to calibrate the camera that only requires the selection of four points on the image that forms a rectangle on the road plane, and two metric references.

First, the radial distortion of the lens must be corrected, to make that imaged lines actually correspond to lines in the road plane. We have applied the well-known second order distortion model, which assumes that a set of collinear points $\{\mathbf{x}_i\}$ are radially distorted by the lens as

$$\mathbf{x}'_i = \mathbf{x}_i(1 + K||\mathbf{x}_i||), \tag{1}$$

where the value of the parameter K can be obtained using five correspondences and applying the Levenberg–Marquardt algorithm.

Next, the calibration of the camera is computed using the road plane to image plane homography. This homography is obtained selecting 4 points in the original image such that these points form a rectangle in the road plane, and applying the DLT algorithm [17]. The resulting homography matrix H can be expressed as

$$H = K \left[\begin{array}{ccc} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t} \end{array} \right], \tag{2}$$

where $\mathbf{r}_1$ and $\mathbf{r}_2$ are the two rotation vectors that define the rotation of the camera (the third rotation vector can be obtained as the cross product $\mathbf{r}_3 = \mathbf{r}_1 \times \mathbf{r}_2$), and $\mathbf{t}$ is the translation vector. If we left multiply Equation 2 by K^{-1} we obtain the rotation and translation directly from the columns of H.

The calibration matrix K can be then found by applying a non-linear optimization procedure that minimizes the reprojection error.

5 Background segmentation and blob extraction

The background segmentation stage extracts those regions of the image that most likely correspond to moving objects. The proposed approach is based on the codewords approach [16] at pixel level.

Given the segmentation, the bounding boxes of blobs with at least a certain area are detected using the approach described in [18]. Then, a recursive process is undertaken to join boxes into larger bounding boxes which satisfy $d_x < t_X$, $d_y < t_Y$, where d_x and d_y are the minimal distances in X and Y from box to box, t_X and t_Y are the corresponding distance thresholds. The recursive process stops when no larger rectangles can be obtained that meet the conditions.

Figure 3 exemplifies the results of the segmentation and blob extraction stages in an image showing two vehicles of different sizes.

6 3D tracking

The 3D tracking stage is fed with the set of observed 2D boxes in the current instant, which we will denote as $\mathbf{z}_t = \{\mathbf{z}_{t,m}\}$, with $m = 1 \ldots M$. Each box is parameterized as $\mathbf{z}_{t,m} = (z_{t,m,x}, z_{t,m,y}, z_{t,m,w}, z_{t,m,h})$ in this domain, i.e. a reference point and a width and height.

The result of the tracking process is the estimate of $\mathbf{x}_t$, which is a vector containing the 3D information of all the vehicles in the scene, i.e. $\mathbf{x}_t = \{\mathbf{x}_{t,n}\}$, with $n = 1 \cdots N_t$, where N is the number of vehicles in the scene at time t, and $\mathbf{x}_{t,n}$ is a vector containing the position, width, height, and length of the 3D box fitting vehicle n.

Using these observations and the predictions of the existing vehicles at the previous time instant, an association data matrix is generated, and used within the observation model and for the detection of entering and exiting vehicles.

The proposed tracking method is based on the probabilistic inference theory, which allows handling the temporal evolution of the elements of the scene, taking into account different types of information (observation, interaction, dynamics, etc.). As a result, we will typically get an estimation of the position and 3D volume of all the vehicles that appear in the observation region of the image (see Fig. 4).

6.1 Bayesian inference

Bayesian inference methods provide an estimation of $p(\mathbf{x}_t|Z^t)$, the posterior density distribution of state $\mathbf{x}_t$, which is the parameterization of the existing vehicles in the scene, given all the estimations up to current time, Z^t.

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

$$p(\mathbf{x}_t|Z^t) = kp(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t|Z^{t-1}), \tag{3}$$

where $p(\mathbf{z}_t|\mathbf{x}_t)$ is the likelihood function that models how likely the measurement $\mathbf{z}_t$ would be observed given the system state vector $\mathbf{x}_t$, and $p(\mathbf{x}_t|Z^{t-1})$ 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 the Kolmogorov–Chapman equation [14]

$$p(\mathbf{x}_t|Z^{t-1}) = \int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|Z^{t-1})d\mathbf{x}_{t-1}. \tag{4}$$

If we hypothesize that the posterior can be expressed as a set of samples

$$p(\mathbf{x}_{t-1}|Z^{t-1}) \approx \frac{1}{N_s}\sum_{i=1}^{N_s}\delta(\mathbf{x}_{t-1} - \mathbf{x}_{t-1}^{(i)}), \tag{5}$$

then

$$p(\mathbf{x}_t|Z^{t-1}) \approx \frac{1}{N_s}\sum_{i=1}^{N_s} p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)}). \tag{6}$$

Therefore, we can directly sample from the posterior distribution since we have its approximate analytic expression [13]:

$$p(\mathbf{x}_t|Z^t) \propto p(\mathbf{z}_t|\mathbf{x}_t)\sum_{i=1}^{N_s} p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)}). \tag{7}$$

An MRF factor can be included to the computation of the posterior to model the interaction between the different elements of the state vector. The MRF factors can be easily inserted into the formulation of the posterior density, since they do not depend on previous time instants [13]. This way, the expression of the posterior density shown in (7), is now rewritten as

$$p(\mathbf{x}_t|Z^t) \propto p(\mathbf{z}_t|\mathbf{x}_t) \prod_{n,n'} \varPhi(\mathbf{x}_{t,n}, \mathbf{x}_{t,n'}) \sum_{i=1}^{N_s} p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)}), \tag{8}$$

where $\varPhi(\cdot)$ is a function that governs the interaction between two elements n and n' of the state vector.

Particle filters are tools that generate this set of samples and the corresponding estimation of the posterior distribution. Although there are many different alternatives, MCMC-based particle filters have been shown to obtain the more efficient estimations of the posterior for high-dimensional problems [13] using the Metropolis–Hastings sampling algorithm. Nevertheless, these methods rely on the definition of a Markov chain over the space of states such that the stationary distribution of the chain is equal to the target posterior distribution. In general, a long chain must be used to reach the stationary distribution, which implies the computation of hundreds or thousands of samples.

In this article, we will see that a much more efficient approach can be used by substituting the Metropolis–Hastings sampling strategy by a line search approach inspired in the slice sampling technique [15].

6.2 Data association

The measurements we got are boxes, typically one per object, although, in some situations, there might be a large box that corresponds to several vehicles (due to occlusions or an undesired merging process in the background subtraction and blob extraction stages), or also a vehicle described by several independent boxes (in case the segmentation suffers fragmentation). For that reason, to define an observation model adapted to this behavior, an additional data association stage is required to link measurements with vehicles. The correspondences can be expressed with a matrix, whose rows correspond to measurements and columns to existing vehicles. Figure 5 illustrates an example data association matrix that will be denoted as D, and Fig. 6 shows some examples of D matrices, corresponding to different typical situations.

The association between 2D boxes with 3D vehicles is carried out by projecting the 3D box into the rectified road domain, and then compute its rectangular hull, that we will denote as $\mathbf{x}'_n$ (let us remove the time index t from here on for the sake of clarity), i.e. the projected version of vehicle $\mathbf{x}_n$. As a rectangular element, this hull is characterized by a reference point and a width and length: $\mathbf{x}'_n = (x'_x, x'_y, x'_w, x'_h)$, analogously to observations $\mathbf{z}_m$. An element $D_{m,n}$ of matrix D is set to one if the observation $\mathbf{z}_m$ intersects with $\mathbf{x}'_n$.

6.3 Observation model

The proposed likelihood model takes into account the data association matrix D, and is defined as the product of the likelihood function associated to each observation, considered as independent:

$$p(\mathbf{z}|\mathbf{x}) = \prod_{m=1}^{M} p(\mathbf{z}_m|\mathbf{x}). \tag{9}$$

Each one of these functions corresponds to a row of matrix D, and is computed as the product of two different types of information:

$$p(\mathbf{z}_m|\mathbf{x}) = p_a(\mathbf{z}_m|\mathbf{x})p_d(\mathbf{z}_m|\mathbf{x}), \tag{10}$$

where $p_a(\cdot)$ is a function relative to the intersection of areas of the 2D observation $\mathbf{z}_m$ and the set of hulls of the projected 3D boxes $\mathbf{x} = \{\mathbf{x}_n\}$ with $n = 1 \dots N$. The second function, $p_d(\cdot)$, is related to the distances between the boxes. Figure 7 illustrates, with several examples, the values of each of these factors and how can they evaluate different $\mathbf{x}'_n$ hypotheses. Figure 8 illustrates these concepts with a simple example of a single observation and a single vehicle hypothesis.

The first function is defined as

$$p_a(\mathbf{z}_m|\mathbf{x}) \propto \exp\left(\frac{\sum_{n=1}^{N} a_{m,n}}{a_m} \frac{\sum_{n=1}^{N} a_{m,n}}{N_m \sum_{n=1}^{N} \omega_{m,n} a_n} \right), \tag{11}$$

where $a_{m,n}$ is the intersection between the 2D box, $\mathbf{z}_m$, and the hull of the projected 3D box, $\mathbf{x}'_n$; a_m and a_n are, respectively, the areas of $\mathbf{z}_m$ and $\mathbf{x}'_n$, and N_m is the number of objects that are associated with observation m according to D. The value $\omega_{m,n}$ is used to weight the contribution of each vehicle:

$$\omega_{m,n} = \frac{a_n}{\sum_{n=1}^{N} a_n} \tag{12}$$

such that $\omega_{m,n}$ ranges between 0 and 1 (it is 0 if object n does actually not intersect with observation m, and 1 if object n is the only object associated to observation m).

The first ratio of Equation 11 represents how much area of observation m intersects with its associated objects. The second ratio expresses how much area of the associated objects intersects with the given observation. Since objects might be as well associated to other observations, the sum of their areas is weighted according to the amount of intersection they have with other observations. After the application of the exponential, this factor tends to return low values if the match between the observation and its objects is not accurate, and high if the fit is correct. Some examples of the behavior of these ratios are depicted in Fig. 7. For instance, the first case (two upper rows) represents a single observation, and two different hypothesized $\mathbf{x}'_n$. It is clear from the figure that the upper-most case is a better hypothesis, and that the area of the observation covered by the hypothesis is larger. Therefore, the first ratio of Equation 11 is 0.86 and 0.72 for the second hypothesis. Analogously, it can be observed that the second ratio indeed represents how much area of the hypothesis is covered by the observation. In this case, the first hypothesis gets 0.77 and the second 0.48. As a result, the value of $p_a(\cdot)$ represents well how the 2D boxes $\mathbf{z}_m$ and $\mathbf{x}'_m$ coincide. The other examples of Fig. 7 show the same behavior for this factor in different configurations.

The factor related to the distances between boxes, $p_d(\cdot)$, computes how aligned is the projection of the 3D objects with their associated observations:

$$p_d(\mathbf{z}_m|\mathbf{x}) \propto \exp\left(-\lambda\left(d_{m,x} + d_{m,y}\right)\right), \tag{13}$$

where $d_{m,x}$ and $d_{m,y}$ are, respectively, the reference distances between the boxes. According to the situation of the vehicle in the scene, these distances are computed in a different manner. For instance, when the vehicle is completely observable in the scene (i.e. it is not entering or leaving), the distance $d_{m,x}$ is computed as

$$d_{m,x} = \frac{\sum_{n=1}^{N} D_{m,n}\left|x'_{n,x} - z_{m,x}\right|}{\sum_{n=1}^{N} D_{m,n}}. \tag{14}$$

The distance in y is defined analogously. This way, the object hypotheses that are more centered on the associated observation obtain higher values of $p_d(\cdot)$. In case the vehicle is leaving, the observation of the vehicle in the rectified view is only partial, and thus this factor is adapted to return high values if the visible end of the vehicle fits well with the observation. In this case, $d_{m,x}$ is redefined as

$$d_{m,x} = \frac{\sum_{n=1}^{N} D_{m,n} \left|(x'_{n,x} + x'_{n,w}) - (z_{m,x} + z_{m,w})\right|}{\sum_{n=1}^{N} D_{m,n}}. \tag{15}$$

Figure 7 depicts as well some examples of the values retrieved by function $p_d(\cdot)$ in some illustrative examples. For instance, consider again the first example (two upper rows): the alignment in x of the first hypothesis is much better, since the centers of the boxes are very close, while the second hypothesis is not well aligned in this dimension. As a consequence, the values of d_x are, respectively, 0.04 and 1.12, which imply that the first hypothesis obtains a higher value of $p_d(\cdot)$. The other examples show some other cases in which the alignment makes the difference between the hypotheses.

The combined effect of these two factors is that the hypotheses whose 2D projections best fit to the existing observations obtain higher likelihood values, taking into account both that the area of the intersection is large, and that the boxes are aligned in the two dimensions of the plane.

6.4 Prior model

The information that we have at time t prior to the arrival of a new observation is related to two different issues: on the one hand, there are some physical restrictions on the speed and trajectory of the vehicles, and, on the other hand, there are some width–length–height configurations more probable than others.

6.4.1 Motion prior

For the motion prior model, we will use a lineal constant-velocity model [19], such that we can perform predictions of the position of the vehicles from $t-1$ to t according to their estimated velocities (at each spatial dimension, x and y).

Specifically, $p(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(A\mathbf{x}_{t-1}|\Sigma)$, where matrix A is a linear matrix that propagates state $\mathbf{x}_{t-1}$ to $\mathbf{x}_t$ with a constant-velocity model [19], and $\mathcal{N}(\cdot)$ represents a multivariate normal distribution.

In general terms, we have observed that within this type of scenarios, this model predicts correctly the movement of vehicles observed from the camera's view point, and is as well able to absorb small to medium instantaneous variations of speed.

6.4.2 Model prior

Since what we want to model are vehicles, the possible values of the tuple WHL (width, height, and length) must satisfy some restrictions imposed by the typical vehicle designs. For instance, it is very unlikely to have a vehicle with width and length equal to 0.5 and 3 m high.

Nevertheless, there is a wide enough variety of possible configurations of WHL such that it is not reasonable to fit the observations to a discrete number of fixed configurations. For that reason, we have defined a flexible procedure that uses a discrete number of models as a reference to evaluate how realistic a hypothesis is. Specifically, we will test how close is a hypothesis to the closest model in the WHL space. If it is close, then the model prior will be high, and low otherwise.

Provided the set of models $\mathcal{X} = \{\mathbf{x}_c\}$, with $c = 1 \dots C$, the expression of the prior is $p(\mathbf{x}_t|\mathcal{X}) = p(\mathbf{x}_t|\mathbf{x}_{c'})$, where $\mathbf{x}_{c'}$ is the model that is closer to $\mathbf{x}_t$. Hence, $p(\mathbf{x}_t|\mathbf{x}_{c'}) = \mathcal{N}(\mathbf{x}_c|\Sigma)$ is the function that describes the probability of a hypothesis to correspond to model $\mathbf{x}_{c'}$. The covariance Σ can be chosen to define how much restrictive is the prior term. If it is set too high, then the impact of $p(\mathbf{x}_t|\mathcal{X}_c)$ on $p(\mathbf{x}_t|\mathbf{z}_t)$ could be negligible, while a too low value could make that $p(\mathbf{x}_t|\mathbf{z}_t)$ is excessively peaked so that sampling could be biased.

In practice, we have used the set of models illustrated in Fig. 9. The number of models and the differences between them depends on how much restrictive we would like to be with the type of vehicles to detect. If we define just a couple of vehicles, or a single static vehicle, then detection and tracking results will be less accurate.

6.5 MRF interaction model

Provided our method considers multiple vehicles within the state vector $\mathbf{x}_t$, we can introduce models that govern the interaction between vehicles in the same scene. The use of such information gives more reliability and robustness to the system estimates, since it better models the reality.

Specifically, we use a simple model that avoids estimated vehicles to overlap in space. For that purpose we define an MRF factor, as in Equation 8. The function $\Phi(\cdot)$ can be defined as a function that penalizes hypotheses in which there is a 3D overlap between two or more vehicles.

The MRF factor can then be defined as

$$\varPhi(\mathbf{x}_n, \mathbf{x}_{n'}) = \begin{cases} 0 \text{ if} \cap (\mathbf{x}_n, \mathbf{x}_{n'}) = 0 \\ \\ 1 \text{ otherwise} \end{cases} \tag{16}$$

between any pair of vehicles characterized by $\mathbf{x}_n$ and $\mathbf{x}_{n'}$, where $\cap(\cdot)$ is a function that returns the volume of intersection between two 3D boxes.

6.6 Input/output control

Appearing and disappearing vehicle control is done through the analysis of the data association matrix, D. If an observed 2D box, $\mathbf{z}_m$, is not associated with any existing object $\mathbf{x}_n$, then a new object event is triggered. If this event is repeated in a determined number of consecutive instants, then the state vector is augmented with the parameters of a new vehicle.

Analogously, if an existing object is not associated with any observation according to D, then a delete object event is triggered. If the event is as well repeated in a number of instants, then the corresponding component $\mathbf{x}_n$ of the state vector is removed from the set.

7 Optimization procedure

Particle filters infer a point-estimate as a statistic (typically, the mean) of a set of samples. Consequently, the posterior distribution has to be evaluated at least once per sample. For high-dimensional problems as ours, MCMC-based methods typically require the use of thousands of samples to reach a stationary distribution. This drawback is compounded for importance sampling methods, since the number of required samples increases exponentially with the problem dimension. In this work, we propose a new optimization scheme that directly finds the point-estimate of the posterior distribution. This way, we avoid the step of sample generation and evaluation, and thus the processing load is dramatically decreased. For this purpose we define a technique that combines concepts of the Gibbs sampler and the slice sampler [20]. Given the previous point-estimate $\mathbf{x}_{t-1}^{(*)}$, an optimization procedure is initialized that generates a movement in the space to regions with higher values of the target function (the posterior distribution). The movement is done by the slice sampling algorithm, by defining a slice that delimits the regions with higher function values around the starting point. The generation of the slice for a single dimension is exemplified in Fig. 10. The granularity is given by the step size $\varDelta x$.

Figure 11 illustrates this method in a 2D example function. This procedure is inspired by the Gibbs sampler since a single dimension is selected at a time to perform the movement. Once the slice is defined, a new start point is selected randomly within the slice, and the process is repeated for the next dimension. In Fig. 11, we can see how the first movement moves $\mathbf{x}_{t-1}^{(*)}$ in the x-direction using a slice of width $3\varDelta x$. The second step generates the slice in the y-direction and selects $\mathbf{x}_t^{(0)}$ randomly within the slice. Two more steps lead to the new best estimation of the posterior maximum at time t.

This technique performs as many iterations as necessary to find a stationary point such that its slice is of size zero. As expected, the choice of the step size is critical because too small values would require evaluating the target function too many times to generate the slices, while too high values could potentially lead the search far away from the targeted maximum.

We have designed this method since it provides fast results, typically stopping at the second iteration. Other known methods, like gradient-descent or second-order optimization procedures, have been tested in this context, being much more unstable. The reason is that they greatly depend on the quality of the Jacobian approximation, which, in our problem, introduces too much error and makes the system tend to lose the track.

For a better visualization, let us study how this procedure behaves to optimize the position and volume of a 3D box for a single vehicle. Figure 12 represents two consecutive frames: the initial state vector at the left image, and the result after the optimization procedure at the right image.

Since the vehicle is quite well modeled in the initial state, we can guess that the optimization process will generate movements in the direction of the movement of the vehicle, while making no modifications on the estimation of the width, length, or height. This is illustrated in Fig. 13. As shown, the slice sampling, in the x-dimension finds that the posterior values around the previous estimate are lower. The reason is that the vehicle is moving, in this example, in a straight trajectory without significantly varying its transversal position inside its lane. The movement of the vehicle is therefore more significant in the y-dimension. Hence, the procedure finds a slice around the previous value for which the posterior value is higher. The algorithm then selects the best evaluated point in the slice, which, in the figure, correspond to four positive movements of width Δy. The rest of dimensions (width, height, and length) get as well no movement since there is no better posterior values around the current estimates.

To exemplify the movement in the y-direction, Fig. 14 shows some of the evaluated hypothesis, which increase the y position of the vehicle. As shown, the slice sampling allows evaluating several points in the slice, and selecting as new point-estimate the one with highest posterior value, which is indeed the hypothesis that best fit to the vehicle.

8 Tests and discussion

There are two different types of tests that identify the performance of the proposed system. On the one hand, detection and classification rates, which illustrates how many miss-detections and false alarms the system suffers. On the other hand, efficiency tests of the proposed sampling algorithm, which depicts the number of evaluations of the posterior distribution $p(\mathbf{x}_t|\mathbf{z}_t)$ are required to reach the target detection and classification rates.

8.1 Detection and classification results

Tests have been carried out using six long sequences (1 h in average each one, over 10,000 vehicles in total), four of them obtained from a low-height camera, and the two others from two different perspectives with higher cameras. These sequences have been selected to evaluate the performance of the proposed method in challenging situations, including illumination variation, heavy traffic situations, shadows, rain, etc.

Considering the detection rates, we have counted the number of vehicles that drive through the scene and are undetected by the system (miss-detections or false negative F_N), the number of non-existing detections (false alarms or false positive, F_P), and the ground truth number of vehicles (N). Moreover, we will consider two vehicle categories: light and heavy vehicles. Although images cannot be used to obtain weight information, we deduce it using the length of the vehicles, i.e. a vehicle is considered as light if its length is lower than 6 m, and heavy otherwise. This approximation is motivated by the fact that road operators typically require that vehicles are classified according to their weight. Hence, we will define pairs of statistics for each type of vehicle, i.e. false positive and negative values and total number of light vehicles (F_{PL}, F_{NL}, N_L), and analogous variables for heavy vehicles (F_{PH}, F_{NH}, N_H).

The results of the tests are shown in Table 1. These results show simultaneously the detection quality, and the classification errors. E_{CL} is the number of light vehicles classified as heavy, and E_{CP} is the number of heavy vehicles classified as light. For a better understanding and comparison of the results, we have computed the associated recall and precision values of each sequence and type of vehicle. Hence, we obtain pairs (R_L, P_L) and (R_H, P_H) for each sequence. These values are computed as

$$Recall = \frac{N_L - F_{NL} - F_{PL} - E_{CL}}{N_L}, \tag{17}$$

$$Precision = \frac{N_L - F_{NL} - F_{PL} - E_{CL}}{N_L - F_{NL} + F_{PL} + E_{CL}} \tag{18}$$

and an analogous expression for heavy vehicles. Recall is related to the number of miss-detections, while precision is related to the number of false alarms.

Figure 15 shows the obtained results. Besides, an example image of each sequence is shown in Fig. 16. As shown, the recall and precision values are all comprised between 80 and 99%, corresponding in all cases the worsen values to the heavy vehicle category. This is due to the wider variety of heavy vehicle sizes, which also causes more problems to the system due to their projected shadows, or the occlusions they generate. For the low-height camera sequences, large vehicles sometimes occupy a very significant part of the image, making that the camera adjust its internal illumination parameters, which causes subsequent detection distortions.

Nevertheless, we have obtained good detection and classification results in all these challenging situations, being of special interest the ability of the system to reliably count vehicles with heavy traffic (such as in the third sequence). The system is also able to work with different type of perspectives, since it computes the calibration of the camera and thus considers the 3D volume of vehicles instead of just 2D silhouettes. The last sequence (Color noise) has been selected since it corresponds to a sequence captured with a low cost camera, which indeed shows significant color noise in some regions of the image. The segmentation and blob generation stages absorb this type of distortion and makes that the detection and classification results are both excellent.

8.2 Sampling results

This subsection shows some experimental results that illustrate the benefits of using the proposed sampling strategy within the Bayesian framework. First, we show with a real example that the proposed method can be used to reach high values of posterior probability with few iterations. Second, we compare the performance of this sampling strategy with that of well known sampling methods typically used in the context of particle filtering and Bayesian inference.

8.2.1 Real data example

The proposed method performance has been evaluated as well according to the number of required evaluations of the posterior distribution to reach the above-mentioned detection and classification rates.

As explained along the article, the proposed sampling strategy allows adapting the number of evaluations to the movement of the vehicle. Hence, typically it is only needed to carry out movements in the y-direction, while the movements in width, height, and length are only necessary in entering and leaving situations.

The system generates a number of samples adapted to the number of vehicles of the scene at each instant. The greater the number of vehicles the greater dimension of the state vector and number of posterior evaluations.

Figure 17 shows the behavior of the system regarding the number of evaluations according to the number of tracked vehicles. We have used accumulated values of different sequences, divided into three characteristic scenarios: low traffic, normal traffic and heavy traffic. The histograms of the left column show the distribution of the number of vehicles for these scenarios, while the right column shows the corresponding distribution of number of evaluations. As shown, the number of objects in the low-traffic scenario does not typically exceed three vehicles simultaneously in the scene, and includes a large amount of instants in which there are no vehicles at all. Therefore, we observe that the system performs a proportional number of evaluations, 50 in average without considering the bin at 0, which correspond to those instants without vehicles.

In the two other situations: normal traffic and heavy traffic, the number of vehicles is increased, and there are some instants with 4 and 5 vehicles in the scene, which requires a higher computational load to the system. The histograms of the number of evaluations show that, in these situations, the number of evaluations ranges between 0 and 100, and between 0 and 200, respectively.

8.2.2 Synthetic data experiments

The following experiments aim to show that the slice sampling-based strategy generates better estimates of a target posterior distribution compared to the importance re-sampling algorithm [14] and the Metropolis–Hastings algorithm.

The tests are carried out as follows. For the sake of simplicity, a target distribution is defined as a multi-variate normal distribution, $N(\mu, \Sigma)$, of D dimension, where $\mu \in R^D$ and $\Sigma \in R^{D \times D}$. The three-mentioned algorithms are executed to generate a number of samples of this target distribution. The error is computed as the norm of the difference between the average value of the samples and the mode of the multi-variate normal distribution $\epsilon = \|\mu - \frac{1}{N}\sum_{n=1}^{N} \mathbf{x}_n\|$, where N is the number of samples, and $\mathbf{x}_n \in R^D$ is the nth sample.

Each algorithm is executed 100 times, and the error is averaged to avoid numerical instability. The test is executed for example instances of the multivariate distribution, where $D = 1, 2, 4, 10$ and asking the algorithms to generate 10 to 1,000 samples.

Figure 18 shows the obtained error of each method according to the number of samples, for 1D, 2D, 4D, and 10D. For low dimensionality (1D, 2D), the importance sampling algorithm performs well, similarly to the slice sampling. The MH algorithm performs well although carefully selecting the step size. We can see that a step size too small makes that the algorithm obtains high rejection rates that affect to the accuracy of the estimation. When the dimensionality of the problem grows (4D or 10D), which is more adapted to real tracking problems, the importance sampling algorithm begins to offer very poor results. The reason is that this method is known to require an exponentially growing number of samples to reach good estimations [13]. We run these tests for obtaining up to 1,000 samples, which is clearly insufficient.

In these high-dimensionality examples, we can see that the performance of the slice sampling-based algorithm is very high, and better than the one of the MH. It is noteworthy that the step size is very important for the MH algorithm, while the SS algorithm adapts the step size to the target function and thus do not require that fine parameter tuning. When the number of samples is low, this drawback makes the MH to fail to reach the regions of the target distribution with higher probability, and thus the error is too large. This is illustrated in Fig. 19, where the MH and the SS methods are compared in a 2D example, using 10 and 100 samples. As shown, the slice-based method reaches the high-probability mass of the target distribution in a couple of iterations while the MH do not. When the number of samples is increased to 100, the MH reaches as well that regions of the space.

Therefore, we can say that, compared to other methods, the SS algorithm (i) generates better estimations with less number of samples; (ii) provides more accurate results; and (iii) is less sensitive to parameter tuning. In summary, the proposed scheme can be used for real applications as the one described in the text which require accurate results and real-time processing, since it can generates good estimates using a reduced number of samples.

8.3 Computation requirements

Finally, attending to the computation time of the whole system implementation, it runs at around 30 fps using images downsampled to 320×240 pixels for processing on an Intel Core2 Quad CPU Q8400 at 2.66 GHz, with 3GB RAM and a NVIDIA 9600GT. This is an industrial PC that satisfies the installation requirements and allows us to process the images in real time.

The program has been implemented in C/C++, using OpenCV primitives for data structure and basic image processing operations, OpenGL for visualization of results, and OpenMP and CUDA for multi-core and GPU programming, respectively.

9 Conclusions

In this article, we have presented the results of the work done in the design, implementation, and evaluation of a vision system designed to represent a serious alternative, cheap, and effective to systems based on other types of sensors in vehicle counting and classification for free flow and shadow tolling applications.

For this purpose, we have presented a method that exploits different information sources and combines them into a powerful probabilistic framework, inspired by the MCMC-based particle filters. Our main contribution is the proposal of a novel sampling system that adapts to the needs of each situation, so that allows for very robust and precise estimates with a much smaller number of point-estimates with respect to other sampling methods such as Importance sampling or the Metropolis–Hastings.

An extensive testing and evaluation phase has led us to collect data on system performance in many situations. We have shown that the system can detect, track, and classify vehicles with very high levels of accuracy, even in challenging situations, including heavy traffic conditions, presence of shadows, rain, and variable illumination conditions.

10 Competing interests

The authors declare that they have no competing interests.

Acknowledgments

This work was partially supported by the Basque Government under the ETORGAI strategic project iToll.

References

1. M Haag, HH Nagel, Incremental recognition of traffic situations from video image sequences. Image and Vision Computing 18:137–153 (2000).
2. B Coifman, D Beymer, P McLauchlan, A real-time computer vision system for vehicle tracking and tracking surveillance. Transportation Research Part C: Emerging Technologies 6:271–288 (1998).
3. NK Kanhere, SJ Pundlik, ST Birchfield, Vehicle segmentation and tracking from a low-angle off-axis camera. In: IEEE Proc. Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 1152–1157 (2005).
4. L Vibha, M Venkatesha, GR Prasanth, N Suhas, PD Shenoy, KR Venugopal, LM Patnaik, Moving vehicle identification using background registration technique for traffic surveillance. In: Proc. of the Int. MultiConference of Engineers and Computer Scientists (2008).

5. C Maduro, K Batista, P Peixoto, J Batista, Estimation of vehicle velocity and traffic intensity using rectified images. In: IEEE International Conference on Image Processing, pp. 777–780 (2008).
6. N Buch, J Orwell, SA Velastin, Urban road user detection and classification using 3D wire frame models. IET Computer Vision Journal 4(2):105–116 (2010).
7. C Pang, W Lam, N Yung, A method for vehicle count in the presence of multiple occlusions in traffic images. IEEE Transactions on Intelligent Transportation Systems 8(3):441–459 (2007).
8. F Bardet, T Chateau, MCMC particle filter for real-time visual tracking. In: IEEE International Conference on Intelligent Transportation Systems, pp. 539–544 (2008).
9. B Johansson, J Wiklund, P Forssén, G Granlund, Combining shadow detection and simulation for estimation of vehicle size and position. Pattern Recognition Letters 30:751–759 (2009).
10. X Zou, D Li, J Liu, Real-time vehicles tracking based on Kalman filter in an ITS. In: International Symposium on Photoelectronic Detection and Imaging, vol SPIE 6623, pp. 662306 (2008).
11. PLM Bouttefroy, A Bouzerdoum, SL Phung, A Beghdadi, Vehicle tracking by non-drifting Mean-Shift using projective Kalman filter. In: IEEE Proc. Intelligent Transportation Systems, pp. 61–66 (2008).
12. X Song, R Nevatia, Detection and tracking of moving vehicles in crowded scenes. In: IEEE Workshop on Motion and Video Computing, pp. 4–8 (2007).
13. Z Khan, T Balch, F Dellaert, MCMC-based particle filtering for tracking a variable number of interacting targets. IEEE Trans. on Pattern Analysis and Machine Intelligence 27(11):1805–1819 (2005).
14. MS Arulampalam, S Maskell, N Gordon, T Clapp, A tutorial on particle filters for online Nonlinear/Non-Gaussian Bayesian tracking. IEEE Trans. on Signal Processing 50(2):174–188 (2002).
15. CM Bishop, Pattern Recognition and Machine Learning (Information Science and Statistics), Springer (2006).
16. K Kim, TH Chalidabhongse, D Harwood, L Davis, Real-time foreground-background segmentation using codebook model. Real-time Imaging 11(3):167–256.

17. RI Hartley, A Zisserman, Multiple view geometry in computer vision. Cambridge University Press (2004).

18. S Suzuki, K Abe, Topological structural analysis of digital binary images by border following. Computer Vision, Graphics and Image Processing 30(1):32–46.

19. PS Maybeck, Stochastic models, estimation, and control, Mathematics in Science and Engineering vol 141. Academic Press, New York, San Francisco, London (1979).

20. R Neal, Slice sampling. Annals of Statistics 31:705–767.

Table 1 Detection and classification results

Sequence	E_{CL}	E_{CP}	F_{NL}	F_{PL}	F_{NP}	F_{PP}	N_L	N_P	R_L	P_L	R_P	P_P
Dusk	0	5	24	9	1	0	1662	118	0.9801	0.9861	0.9492	0.9573
Rain and shadow	33	26	73	88	7	11	4516	627	0.9570	0.9484	0.9298	0.8780
Traffic jam	10	28	63	7	16	4	4796	563	0.9833	0.9891	0.9147	0.9180
Dusk and rain	8	13	19	48	0	1	968	115	0.9225	0.8842	0.8783	0.8145
Perspective	2	10	30	18	2	2	614	101	0.9186	0.9216	0.8614	0.8447
Color noise	0	3	0	1	0	0	561	23	0.9982	0.9912	0.8696	0.8696

Fig. 1 Block diagram of the vision-part of the system.

Fig. 2 Two different viewpoints generate different perspective distortion: **(a)** synthetic example of a vehicle and the road observed with a camera installed in a pole; and **(b)** installed in a gate.

Fig. 3 Vehicle tracking with a rectangular vehicle model. *Dark boxes* correspond to blob candidates, *light* to previous vehicle box and *white* to the current vehicle box.

Fig. 4 Tracking example: The *upper row* shows the rendering of the obtained 3D model of each vehicle. As shown, the appearance and disappearance of vehicles is handled by means of an entering and exiting region, which limits the road stretch that is visualized in the rectified domain (*bottom row*).

Fig. 5 Association of measurements $\mathbf{z}_{t,m}$ with existing objects $\mathbf{x}_{t-1,n}$, and the corresponding data association matrix D (measurements correspond to the row of D and objects to the columns).

Fig. 6 Different simple configurations of the data association matrix and their corresponding synthetic vehicles projections (in *blue*), and measurements (in *red*).

Fig. 7 Example likelihood for three different scenes (grouped as pairs of rows). For each one, two $\mathbf{x}$ hypotheses are proposed and the associated likelihood computed. In *red*, the observed 2D box, and in *blue*, the projected 3D boxes of the vehicles contained in $\mathbf{x}$.

Fig. 8 Likelihood example: (a) a single observation (2D bounding box); **(b)** a single vehicle hypothesis, where the 3D vehicle is projected into the rectified view (in *solid lines*), and its associated 2D bounding box is shown in *dashed lines*; (c) the relative distance between the 2D boxes $(d_{m,x}, d_{m,y})$, and the intersection area $a_{m,n}$.

Fig. 9 Example set of 3D box models, $\mathcal{X}$, comprising small vehicles like cars or motorbikes, and long vehicles like buses and trucks.

Fig. 10 This illustration depicts a single movement from a start point $x^{(i)}$ to a new position $x^{(i+1)}$ in a single dimension by creating a slice.

Fig. 11 Example execution of the proposed optimization procedure on a 2D synthetic example, showing three iterations.

Fig. 12 Example optimization procedure between two frames.

Fig. 13 Movement at each dimension for example case shown in Fig. 12. As shown, only the slice in the y dimension shows movements that increase the value of $p(\mathbf{x}_t|\mathbf{z}_t)$. For simplicity, the step size is the same for all dimensions (since all of them represent the same magnitude: meters).

Fig. 14 Linear movement in y, and their associated $p(\mathbf{x}_t|\mathbf{z}_t)$ values.

Fig. 15 Recall and precision graphs for the different sequences defined in Table 1. The values of the graph for each sequence correspond to the recall–precision pairs of light (*left*) and heavy (*right*) vehicles.

Fig. 16 Example results for each one of the sequences used for testing. From *left* to *right*, the sequences correspond to those indexed in Table 1.

Fig. 17 Distribution of number of objects (*left*) and number of evaluations of the posterior (*right*) for three different traffic scenarios.

Fig. 18 Comparison of the performance of the slice-based sampling method, the importance sampling and the Metropolis–Hastings algorithms. IRS, importance resampling; MH (σ), Metropolis–Hastings with Gaussian proposal distribution with standard deviation σ; SS, slice sampling-based approach.

Fig. 19 Comparison of the performance of Metropolis–Hastings and the proposed slice-based method when using 10 and 100 samples in a 2D target function.