Automatic vehicle counting system for traffic monitoring

Alain Crouzil,[a,*] Louahdi Khoudour,[b] Paul Valiere,[c] and Dung Nghy Truong Cong[d]
[a]Université Paul Sabatier, Institut de Recherche en Informatique de Toulouse, 118 route de Narbonne, 31062 Toulouse Cedex 9, France
[b]Center for Technical Studies of South West, ZELT Group, 1 avenue du Colonel Roche, 31400 Toulouse, France
[c]Sopra Steria, 1 Avenue André-Marie Ampère, 31770 Colomiers, France
[d]Ho Chi Minh City University of Technology, 268 Ly Thuong Kiet Street, 10th District, Ho Chi Minh City, Vietnam

Abstract. The article is dedicated to the presentation of a vision-based system for road vehicle counting and classification. The system is able to achieve counting with a very good accuracy even in difficult scenarios linked to occlusions and/or presence of shadows. The principle of the system is to use already installed cameras in road networks without any additional calibration procedure. We propose a robust segmentation algorithm that detects foreground pixels corresponding to moving vehicles. First, the approach models each pixel of the background with an adaptive Gaussian distribution. This model is coupled with a motion detection procedure, which allows correctly location of moving vehicles in space and time. The nature of trials carried out, including peak periods and various vehicle types, leads to an increase of occlusions between cars and between cars and trucks. A specific method for severe occlusion detection, based on the notion of solidity, has been carried out and tested. Furthermore, the method developed in this work is capable of managing shadows with high resolution. The related algorithm has been tested and compared to a classical method. Experimental results based on four large datasets show that our method can count and classify vehicles in real time with a high level of performance (>98%) under different environmental situations, thus performing better than the conventional inductive loop detectors.

Keywords: computer vision; tracking; traffic image analysis; traffic information systems.

1 Introduction

A considerable number of technologies able to measure traffic flows are available in the literature. Three of the most established ones are summarized below.

Inductive loops detectors (ILD): The most deployed are inductive loops installed on roads all over the world.[1] This kind of sensor presents some limitations linked to the following factors: electromagnetic fields, vehicles moving very slowly not taken into account (<5 km/h), vehicles close to each other, and very small vehicles. Furthermore, the cost for installation and maintenance is very high.

Infrared detectors (IRDs): There are two main families among the IRDs: passive IR sensors and active ones (emission and reception of a signal). This kind of sensor presents low accuracy in terms of speed and flow. Furthermore, the active IRDs do not allow detecting certain vehicles such as two-wheeled or dark vehicles. They are also very susceptible to rain.[1]

Laser sensors: Laser sensors are applied to detect vehicles, to measure the distance between the sensor and the vehicles, and the speed and shape of the vehicles. This kind of sensor does not allow detecting fast vehicles, is susceptible to rain, and presents difficulty in detecting two-wheeled vehicles.[1]

A vision-based system is chosen here for several reasons: the quality of data is much richer and more complete compared to the information coming from radar, ILD, or lasers. Furthermore, the computational power of contemporary computers is able to meet the requirements of image processing.

In the literature, a great number of methods dealing with vehicle classification using computer vision can be found. In fact, the tools developed in this area are either industrial systems developed by companies like Citilog in France,[2] or FLIR Systems, Inc.,[3] or specific algorithms developed by academic researchers. According to Ref. 4, many commercially available vision-based systems rely on simple processing algorithms, such as virtual detectors, in a way similar to ILD systems, with limited vehicle classification capabilities, in contrast to more sophisticated academic developments.[5,6]

This study presents the description of a vision-based system to automatically obtain traffic flow data. This system operates in real time and can work during challenging scenarios in terms of weather conditions, with very low-cost cameras, poor illumination, and in the presence of many shadows. In addition, the system is conceived to work on the already existing cameras installed by the transport operators. Contemporary cameras are used for traffic surveillance or detection capabilities like incident detections (counterflow, stopped vehicles, and so on). The objective in this work is to directly use the existing cameras without changing existing parameters (orientation, focal lens, height, and so on). From a user-needs analysis carried out with transport operators, the system presented here is mainly dedicated to a vehicle counting and classification for ring roads (cf. Fig. 1).

Recently, Unzueta et al.[7] published a study on the same subject. The novelty of their approach relies on a multi-cue background subtraction procedure in which the segmentation thresholds adapt robustly to illumination changes. Even if the results are very promising, the datasets used in the evaluation

*Address all correspondence to: Alain Crouzil, E-mail: alain.crouzil@irit.fr

phase are very limited (duration of 5 min.). Furthermore, the handling of severe occlusions is out of the scope of his paper.

The novelty of our approach is threefold. (1) We propose an approach for background subtraction, derived from improved Gaussian mixture models (GMMs), in which the update of the background is achieved recursively. This approach is combined with a motion detection procedure, which can adapt robustly to illumination changes, maintaining a high sensitivity to new incoming foreground objects. (2) We also propose an algorithm able to deal with strong, moving casted shadows. One of the evaluation datasets is specifically shadow-oriented. (3) Finally, a new algorithm able to tackle the problems raised by severe occlusions among cars, and between cars and trucks is proposed.

We include experimental results with varying weather conditions, on sunny days with moving directional shadows and heavy traffic. We obtain vehicle counting and classification results much better than those of ILD systems, which are currently the most widely used systems for these types of traffic measurements, while keeping the main advantages of vision-based systems, i.e., not requiring the cumbersome operation or installation of equipment at the roadside or the need for additional technology such as laser scanners, tags, or GPS.

2 Related Work

Robust background subtraction, shadows management, and occlusion care are the three main scientific contributions of our work.

2.1 *Background Subtraction*

The main aim of this section is to provide a brief summary of the state-of-the-art moving object detection methods based on a reference image. The existing methods of background subtraction can be divided according to two categories:[7] nonparametric and parametric methods. Parametric approaches use a series of parameters that determines the characteristics of the statistical functions of the model, whereas nonparametric approaches automate the selection of the model parameters as a function of the observed data during training.

2.1.1 *Nonparametric methods*

The classification procedure is generally divided into two parts: a training period of time and a detection period. The nonparametric methods are efficient when the training period is sufficiently long. During this period, the setting up of a background model consists in saving the possible states of a pixel (intensity, color, and so on).

Median value model. This adaptive model was developed by Greenhill et al. in Ref. 8 for moving objects extraction during degraded illumination changes. Referring to the different states of each pixel during a training period, a background model is thus elaborated. The background is continuously updated for every new frame so that a vector of the median values (intensities, color, and so on) is built from the $N/2$ last frames, where N is the number of frames used during the training period. The classification background/object is simply obtained by thresholding the distance between the value of the pixel to classify and its counterpart in the background model. In order to take into account the illumination changes, the threshold considers the width of the interval containing the pixel values.

This method based on the median operator is more robust than that based on running average.

Codebook. The codebook method is the most famous nonparametric method. In Ref. 9, Kim et al. suggest modeling the background based on a sequence of observations of each pixel during a period of several minutes. Then, similar occurrences of a given pixel are represented according to a vector called codeword. Two codewords are considered as different if the distance, in the vectorial space, exceeds a given threshold. A codebook, which is a set of codewords, is built for every pixel. The classification background/object is based on a simple difference between the current value of each pixel and each of the corresponding codewords.

2.1.2 *Parametric methods*

Most of the moving objects extraction methods are based on the temporal evolution of each pixel of the image. A sequence of frames is used to build a background model for every pixel. Intensity, color, or some texture characteristics could be used for the pixel. The detection process consists in independently classifying every pixel in the object/background classes, according to the current observations.

Gaussian model. In Ref. 10, Wren et al. suggest to adapt the threshold on each pixel by modeling the intensity distribution for every pixel with a Gaussian distribution. This model could adapt to slow changes in the scene, like progressive illumination changes. The background is updated recursively thanks to an adaptive filter. Different extensions of this model were developed by changing the characteristics at pixel level. Gordon et al.[11] represent each pixel with four components: the three color components and the depth.

Gaussian mixture model. An improvement of the previous model consists in modeling the temporal evolution with a GMM. Stauffer and Grimson[12,13] model the color of each pixel with a Gaussian mixture. The number of Gaussians must be adjusted according to the complexity of the scene. In order to simplify calculations, the covariance matrix is considered as diagonal because the three color channels are taken into account independently. The GMM model is updated at each iteration using the k-mean algorithm. Harville et al.[14] suggest to use GMM in a space combining the depth and YUV space. They improve the method by controlling the training rate according to the activity in the scene. However, its response is very sensitive to sudden variations of the background like global illumination changes. A low training rate will produce numerous false detections during an illumination change period, whereas a high training rate will include moving objects in the background model.

Markov model. In order to consider the temporal evolution of a pixel, the order of arrival of the gray levels on this pixel is useful information. A solution consists in modeling the gray level evolution for each pixel by a Markov chain. Rittscher et al.[15] use a Markov chain with three states: object, background, and shadow. All the parameters of the chain, initial, transition, and observation probabilities, are

Fig. 1 Some images shot by the existing CCTV system in suburban fast lanes at Toulouse in the southwest of France.

estimated off-line on a training sequence. Stenger et al.[16] proposed an improvement, since after a short training period, the model of the chain and its parameters continues to be updated. This update, carried out during the detection period, allows us to better deal with the nonstationary states linked, for example, to sudden illumination changes.

2.2 *Shadow Removal*

In the literature, several shadow detection methods exist, and, hereunder, we briefly mention some of them.

In Ref. 17, Grest et al. determine the shadow zones by studying the correlation between a reference image and a current image from two hypotheses. The first one states that a pixel in a shadowed zone is darker than the same pixel in an illuminated zone. The second one starts from a correlation between the texture of a shadowed zone and the same zone of the reference image. The study of Joshi et al.[18] shows correlations between the current image and the background model using four parameters: intensity, color, edges, and texture.

Avery et al.[19] determine the shadow zones with a region-growing method. The starting point is located at the edge of the segmented object. Its position is calculated thanks to the sun position obtained from GPS data and time codes of the sequence.

Song et al.[20] make the motion detection with Markov chain models and detect shadows by adding different shadow models.

Recent methods for both background subtraction and shadow suppression mix multiple cues, such as edges and color, to obtain more accurate segmentations. For instance, Huerta et al.[21] apply heuristic rules by combining a conical model of brightness and chromaticity in the RGB color space along with edge-based background subtraction, obtaining better segmentation results than other previous state-of-the-art approaches. They also point out that adding a higher-level model of vehicles could allow for better results, as these could help with bad segmentation situations. This optimization is seen in Ref. 22, in which the size, position, and orientation of a three-dimensional bounding box of a vehicle, which includes shadow simulation from GPS data, are optimized with respect to the segmented images. Furthermore, it is shown in some examples that this approach can improve the performance compared to using only shadow detection or shadow simulation. Their improvement is most evident when shadow detection or simulation is inaccurate. However, a major drawback for this approach is the initialization of the box, which can lead to severe failures.

Other shadow detection methods are described in recent survey articles.[23,24]

2.3 *Occlusion Management*

Except when the camera is located above the road, with perpendicular viewing to the road surface, when vehicles are close, they partially occlude one another and correct counting is difficult. The problem becomes harder when the occlusion occurs as soon as the vehicles appear in the field of view. Coifman et al.[25] propose tracking vehicle features and to group them by applying a common motion constraint. However, this method fails when two vehicles involved in an occlusion have the same motion. For example, if one vehicle is closely following another, the latter partially occludes the former and the two vehicles can move with the same speed and their trajectory can be quite similar. This situation is usually observed when the traffic is too dense for drivers to keep large spacings between vehicles and to avoid occlusions, but not enough congested to make them constantly change their velocity. Pang et al.[5] propose a threefold method: a deformable model is geometrically fitted onto the occluded vehicles; a contour description model is utilized to describe the contour segments; a resolvability index is assigned to each occluded vehicle. This method provides very promising results in terms of counting capabilities. Nonetheless, the method needs the camera to be calibrated and the process is time-consuming.

3 Moving Vehicle Extraction and Counting

3.1 *Synopsis*

In this work, we have developed a system that automatically detects and counts vehicles. The synopsis of the global process is presented in Fig. 2. The proposed system consists of five main functions: motion detection, shadow removal, occlusion management, vehicle tracking, and trajectory counting.

The input of the system is, for instance, a video footage (in the current version of the system, we use a prerecorded video), while the output of the system is an absolute number of vehicles. The following sections describe the different processing steps of the counting system.

3.2 *Motion Detection*

Motion detection, which provides a classification of the pixels into either foreground or background, is a critical task in many computer vision applications. A common approach to detect moving objects is background subtraction, in which each new frame is compared to the estimated background model.

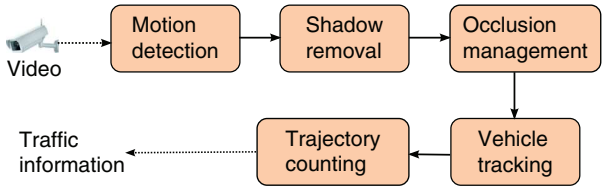

Fig. 2 Synopsis of the proposed system for vehicle counting.

Exterior environment conditions like illumination variations, casted shadows, and occlusions can affect motion detection and lead to wrong counting results. In order to deal with such particular problems, we propose an approach based on an adaptive background subtraction algorithm coupled with a motion detection module. The synopsis of the proposed approach is shown in Fig. 3.

The first two steps, background subtraction and motion detection, are independent and their outputs are combined using the logical AND operator to get the motion detection result. Then, an update operation is carried out. This ultimate step is necessary for motion detection at the next iteration. Those steps are detailed below.

3.2.1 *Background subtraction using Gaussian mixture model*

The GMM method for background subtraction consists in estimating a density function for each pixel. The pixel distribution is modeled as a mixture of N_G Gaussians. The probability of occurrence of a color $I^t(p)$ at the given pixel p is estimated as

$$P[I^t(p)|I_p] = \sum_{i=1}^{N_G} w_i^t(p)\eta[I^t(p)|\mu_i^t(p), \mathbf{\Sigma}_i^t(p)], \tag{1}$$

where $w_i^t(p)$ is the mixing weight of the i'th component at time t, for pixel p ($\sum_{i=1}^{N_G} w_i^t(p) = 1$). Terms $\mu_i^t(p)$ and $\mathbf{\Sigma}_i^t(p)$ are the estimates of the mean and the covariance matrix that describe the i'th Gaussian component. Assuming that the three color components are independent and have the same variances, the covariance matrix is of the form $\mathbf{\Sigma}_i^t(p) = \sigma_i^t(p)I$.

The current pixel p is associated with Gaussian component k if $\|I^t(p) - \mu_k^t(p)\| < S_d\sigma_k^t(p)$, where S_d is a multiplying coefficient of the standard deviation of a given Gaussian. The value of S_d generally lies between 2.5 and 4, depending on the variation of lighting condition of the scene. We fixed it experimentally to 2.7.

For each pixel, the parameters of the matched component k are then updated as follows (the pixel dependence has been omitted for brevity):

$$\begin{cases} \mu_k^t = \left(1 - \frac{\alpha}{w_k^t}\right)\mu_k^{t-1} + \frac{\alpha}{w_k^t} I^t, \\ (\sigma_k^t)^2 = \left(1 - \frac{\alpha}{w_k^t}\right)(\sigma_k^{t-1})^2 + \frac{\alpha}{w_k^t}(I^t - \mu_k^t)^2, \\ w_k^t = (1-\alpha)w_k^{t-1} + \alpha, \end{cases} \tag{2}$$

where $\alpha(p)$ is the updating coefficient of pixel p. An updating matrix that defines the updating coefficient of each pixel will be reestimated at the final stage of the motion detection process.

For the other components that do not satisfy the above condition, their weights are adjusted with

$$w_k^t = (1-\alpha)w_k^{t-1}. \tag{3}$$

If no matched component can be found, the component with the least weight is replaced by a new component with mean $I^t(p)$, an initial variance, and a small weight w_0.

In order to determine whether p is a foreground pixel, all components are first ranked according to the value $w_k^t(p)/\sigma_k^t(p)$. High-rank components, which have low variances and high probabilities, are typical characteristics of background. The first $C(p)$ components describing the background are then selected by the following criterion:

$$C(p) = \arg\min_{C(p)} \left[\sum_{i=1}^{C(p)} w_i^t(p) > S_B\right], \tag{4}$$

where S_B is the rank threshold, which measures the minimum portion of the components that should be accounted for the background. The more complex the background motion, the more the number of Gaussians needed and the higher the value of S_B.

Pixel p is declared as a background pixel if $I^t(p)$ is associated with one of the background components. Otherwise, it is detected as a foreground pixel.

This moving object detection using GMM could also be employed to detect motionless vehicles. Indeed, this functionality dealing with safety is often questioned by transport operators. In our ring road environment, our main concern is to detect and count moving vehicles. Furthermore, we do not consider traffic jam periods because, in this case, the vehicle flow will decrease, and it is more useful to calculate the density of vehicles.

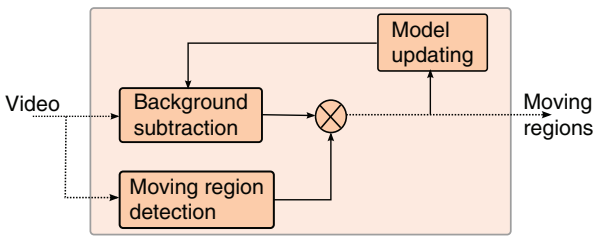

Fig. 3 Synopsis of the motion detection module.

3.2.2 *Moving region detection*

In order to produce better localizations of moving objects and to eliminate all the regions that do not correspond to the foreground, a second algorithm is combined with the GMM method. This algorithm is much faster than the first one and maintains the regions belonging to real moving objects and eliminates noise and false detections. This module looks into the difference among three consecutive frames. This technique has the advantage of requiring very few resources. The binary motion detection mask is defined by

$$M^t(p) = \left[\frac{|I^t(p) - I^{t-1}(p) - \mu_1|}{\sigma_1} > S_M\right] \cup \left[\frac{|I^{t-1}(p) - I^{t-2}(p) - \mu_2|}{\sigma_2} > S_M\right], \tag{5}$$

where $I^t(p)$ is the gray level of pixel p at time t, μ_1 and σ_1 are the mean and the standard deviation of $|I^t - I^{t-1}|$, and S_M is a threshold of the normalized image difference. The value of S_M has been experimentally defined to be 1.0 in our application.

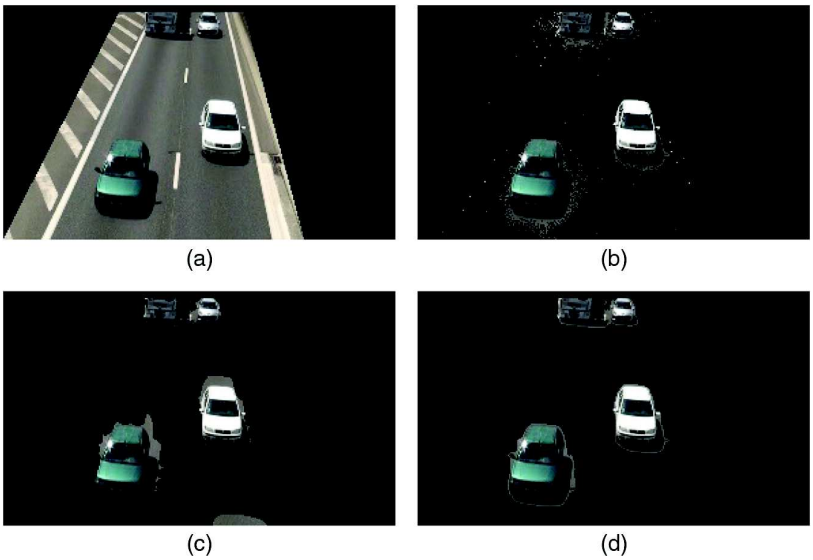

Fig. 4 Combination of the two results: (a) observed scene, (b) foreground detected by GMM, (c) moving region detection result, and (d) final result.

3.2.3 *Result combination and model updating*

At this stage, the results of the GMM and of the moving region detection methods are merged. This leads to moving object detection illustrated by Fig. 4. Figure 4(a) shows the observed scene. In Fig. 4(b), the GMM method has precisely segmented moving objects but noise still remains. The motion region detection [Fig. 4(c)] precisely generates an undesired artifact behind the vehicle, which is eliminated after the combination of the two methods [Fig. 4(d)]. Noise is also eliminated.

The updating matrix that defines the updating coefficient of the Gaussian mixture of each pixel, used in Eqs. (2) and (3), is reestimated at this step. It is a probability matrix that defines the probability for a pixel to be part of the background. Initially, each element of the updating matrix is equal to M. We experimentally defined M to be 0.01 in our application. Then, the coefficients of this matrix are reestimated as follows:

$$\alpha(p) = \begin{cases} m & \text{if } p \text{ is detected as a pixel in motion,} \\ M & \text{otherwise,} \end{cases} \tag{6}$$

where $m \ll M$. We fixed m to 0.0001 in our application.

The algorithm is able to tackle the problems of difficult environments by extracting the moving objects with accuracy thanks to the background subtraction algorithm based on GMM coupled with an adaptive update of the background model, and by managing important illumination changes with the moving region detection module. In Fig. 5, an illustration of the ability of the algorithm to deal with artifacts is

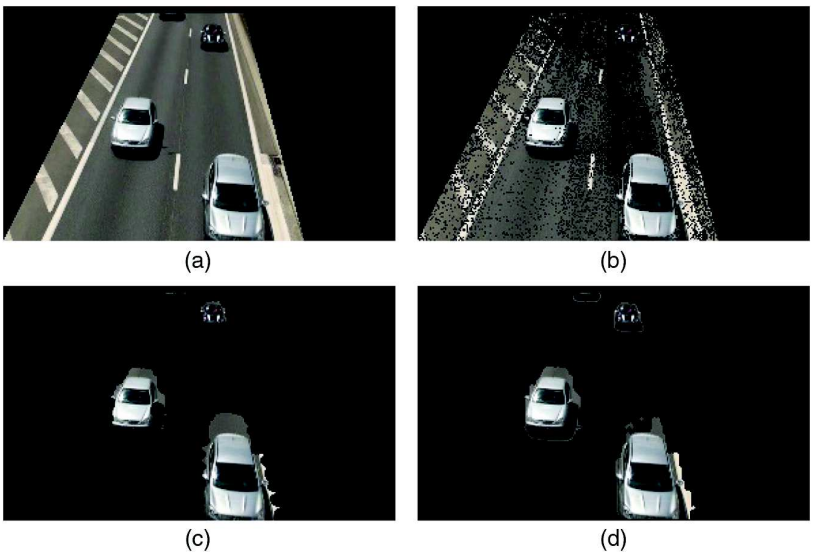

Fig. 5 Background perturbation illustration: (a) observed scene, (b) foreground detected by GMM, (c) moving region detection result, and (d) final result.

provided. Observed scene [Fig. 5(a)] was captured after an important background perturbation was caused by the passing of a truck a few frames earlier. The detected foreground [Fig. 5(b)] is disturbed, but the moving region detection module [Fig. 5(c)] allows us to achieve a satisfying result [Fig. 5(d)].

3.3 *Shadow Elimination*

For shadow elimination, the algorithm developed is inspired from Xiao's approach.[26] This latter was modified and adapted to our problem. The authors have noticed that in a scene including vehicles during a period with high illumination changes, these vehicles present strong edges whereas shadows do not present such marked edges. In fact, from where the scene is captured, road seems to be relatively uniform. In a shadowed region, contrast is reduced and reinforces this characteristic. Edges on the road are located only on marking. On the contrary, vehicles are very textured and contain many edges. Our method aims at correctly removing shadows while preserving the initial edges of the vehicles. As shown in Fig. 6, all steps constituting our method are processed in sequence. Starting from results achieved by the motion detection module, we begin to extract edges. Then, exterior edges are removed. Finally, blobs (regions corresponding to vehicles in motion) are extracted from remaining edges. Each step is detailed in the following paragraphs. This method is efficient, whatever the difficulty linked to the shadow.

3.3.1 *Edge extraction*

Edge detection is a fundamental tool in image processing, which aims at identifying in a digital image pixels corresponding to object contours. We used the Canny's filter,[27] which is an efficient edge detector, with hysteresis thresholding allowing us to detect a sufficient number of edges belonging to the vehicles while maintaining a low number of detected edges on the road. Canny's filter is applied on the foreground regions determined by the last module of motion detection detailed in Sec. 3.2. This foreground image is first dilated with a 3×3 structuring element (SE) to ensure getting all vehicle edges. In our situation, applying the filter on the three RGB channels of the images would not bring significant additional information. That is why we simply use it on a gray-level image. Moreover, it reduces processing time. As shown in Fig. 7, from the observed scene [Fig. 7(a)] and as a result of the moving region detection module [Fig. 7(b)], foreground edges [Fig. 7(c)] are extracted. It can be noticed that shadow areas are linked to vehicles only with their exterior edges.

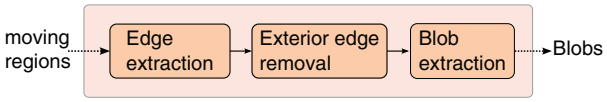

Fig. 6 Synopsis of the shadow removal module.

3.3.2 *Exterior edge removal*

To remove exterior edges, an erosion is applied on the binary image previously dilated. Since the image was dilated with a 3×3 SE, it is now necessary to use a bigger SE to completely eliminate the exterior edges. For that, we apply an erosion operation with a 7×7 SE to remove exterior edges on a two- or three-pixel width. A logical AND is then processed between this eroded image and the previously detected edges. Thus, only interior edges are kept. As illustrated in Fig. 8, from an observed scene in the presence of shadows [Fig. 8(a)] and the detected edges [Fig. 8(b)], this module removes most of the exterior edges [Fig. 8(c)]. The rest will be removed by the next operation, described in the next paragraph.

3.3.3 *Blob extraction*

The goal of this procedure is to extract blobs from the remaining edges. It consists of horizontal and vertical operations, which give two results. For the horizontal operation, we proceed as follows: on each row, the distance in pixels

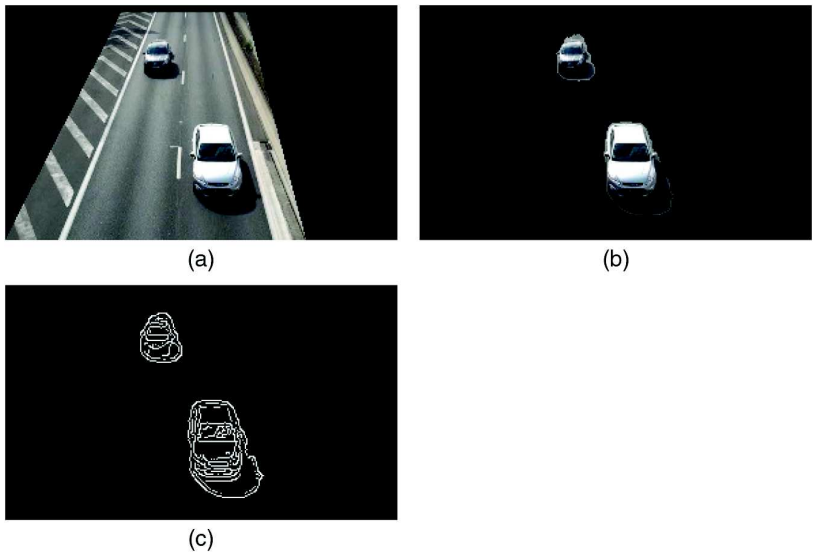

Fig. 7 Foreground edges: (a) observed scene, (b) moving region detection result, and (c) detected edges.

Fig. 8 Exterior edge removal: (a) observed scene, (b) detected edges, and (c) interior edges.

Fig. 9 Shadow elimination and blob extraction: (a) initial vehicle, (b) moving region detection, (c) edge extraction, (d) exterior edge removal, and (e) final blob.

between two edge pixels is computed. If this distance is lower than a threshold, then the pixels between these two points are set to 1. The same operation is made on the columns for the vertical operation. Two different thresholds are chosen, according to vertical or horizontal operation, to eliminate undesired edges from shadows. In our application, we fixed the thresholds experimentally to 5 for horizontal operation and to 17 for the vertical operation.

Then, the two results coming from vertical and horizontal operations are merged. A pseudo-closing is applied to fill small remaining cavities. To remove small asperities, we apply an erosion with a 5×5 SE and finally a dilation with a 7×7 SE. The SE is bigger for the dilation to recover initial edges.

Figure 9 shows an illustration of the whole procedure for shadow elimination and blob extraction.

3.4 *Occlusion Management*

Most existing methods consider cases in which occlusions appear during the sequence but not from the beginning of the sequence. We have developed a new method that can treat occlusions occurring at any time. The first step consists in determining, among all detected blobs, those that potentially contain several vehicles and are candidates to be split. The synopsis of this module is illustrated in Fig. 10.

3.4.1 *Candidate selection*

In order to determine potential candidates among all tracked blobs, we analyze their shapes. Usually, a vehicle is roughly a convex object. If the vehicle is correctly segmented, its shape has only a few cavities. We make the assumption that if a blob is composed of several vehicles, its shape is less convex. Indeed, two convex objects side by side could form a new concave one. The solidity of an object is the object area to convex hull area ratio. It measures the deviation of a shape from being convex. We assume that a blob, corresponding to one vehicle, has a solidity $\geq$90%. Blobs that do not respect this criterion are submitted to the splitting procedure. Jun et al. complete this criterion of solidity in Ref. 28 with eccentricity and orientation. These criteria are quite interesting. However, in our case, in urban highway, vehicle trajectories are mainly rectilinear. So, the criterion of orientation is ineffective here.

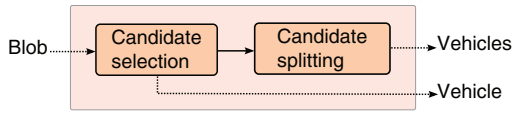

Fig. 10 Synopsis of the occlusion management module.

3.4.2 *Candidate splitting*

We propose to consider the evolution of the blob width along the axis of the road. In our case, the camera is facing the road and the projection of the road axis can be considered as approximately vertical. The blob splitting procedure analyzes the width of the blob on each row of the smallest bounding box of the blob. Figure 11 illustrates the variation of the blob width along the vertical axis showing, on the left side, the binary image of a blob and, on the right side, the width image where the white pixels belonging to the blob have been grouped at the beginning of each row. So the position of the rightmost white pixel represents the width of the blob. As we do not know the number of vehicles in the blob, we begin to separate it into two new blobs. Then, their solidities are calculated and they are recursively segmented, if necessary.

For a blob of height H, all the widths are represented by a vector containing the marginal sums (here, the number of white pixels) along the rows of the binary image of the blob. The blob is split by separating the width vector into two classes. We use the minimum error thresholding (MinError) algorithm proposed by Kittler and Illingworth in Ref. 29. Considering the vector of the width values of the blob as a mixture of two Gaussian distributions, this algorithm calculates the threshold that minimizes the classification error. The returned value is the row splitting the blob into two parts.

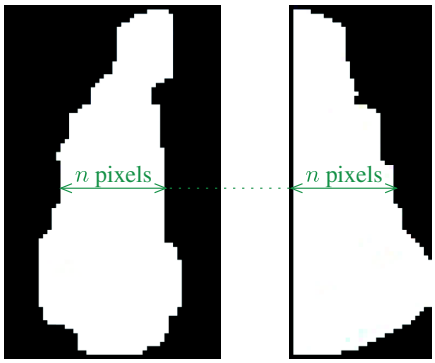

Fig. 11 Variation of the blob width along the vertical axis: (a) binary image of the smallest bounding box of the blob and (b) corresponding width image.

From detected blobs, which are in white in Figs. 12(b) and 12(d), we obtain the splitting results shown in Fig. 13. The two Gaussian curves minimizing the classification error are displayed in red and blue. The corresponding thresholds are represented by green lines.

Occasionally, the iterative MinError algorithm does not converge or converge to a value out of the $[0; H-1]$ interval. When this occurs, only one Gaussian function is appropriate to approximate the blob widths and the blob is not split. It could happen in two cases: (1) it is possible that the occlusion between two vehicles is so strong that the resulting blob might be convex and (2) a vehicle can also be badly segmented and fail the solidity test.

3.5 *Vehicle Tracking*

After the previous module of motion detection, shadow removal, and occlusion management, all blobs do not necessarily match with a single vehicle. Therefore, some artifacts can remain or several blobs can correspond to the same vehicle. A way to overcome this is to consider trajectories. This is what tracking does. It allows counting a vehicle only once. Kalman filter is very well adapted to the kinds of motion in our sequences (rectilinear and smooth). It is a fast filter whose results are accurate enough for our requirements. The algorithm works in a two-step process: in the prediction step, Kalman filter produces

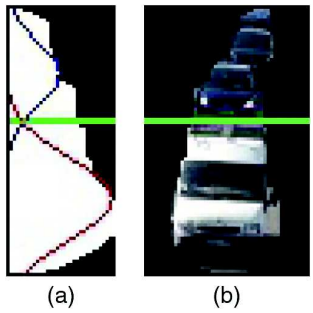

Fig. 13 Blob splitting: (a) result of MinError on the width image and (b) blob image with the corresponding splitting row.

estimates of the current state variables, along with their uncertainties. Once the outcome of the next measurement is observed, these estimates are updated. For each detected blob, a structure is used to save the information about it. All the position states are kept in a vector and a status (counted or not) is used to be sure to count the blob only once. At each iteration, current and previous states are compared to match existing blob or to create new ones. Temporal and definitive trail disappearance is checked. In the case of a temporal disappearance, the trail is kept and its evolution depends on Kalman prediction. A definitive disappearance implies the deletion of the trail. Vehicles are tracked until they disappear from the scene. As the blob states do not abruptly change between two consecutive frames, we have forbidden big changes because they could happen with a bad shadow elimination or with an unexpected fusion of two vehicles. For that, we compare current and previous positions and compute the Euclidean distance between them. If it is greater than a fixed threshold, we use Kalman prediction at the previous state instead of current measure to predict the new state.

3.6 *Trajectory Counting*

First of all, we define a counting zone delimited by two virtual lines. A compromise has to be chosen on its size. This zone has to be large enough to avoid too many false positives and small enough to count every vehicle whatever its size (two-wheelers, small cars, and so on). In our case, we take into account vehicles going on a one-way direction. So, we define a single entry line, which is the upper line, and a single exit line, which is the lower line in Fig. 14.

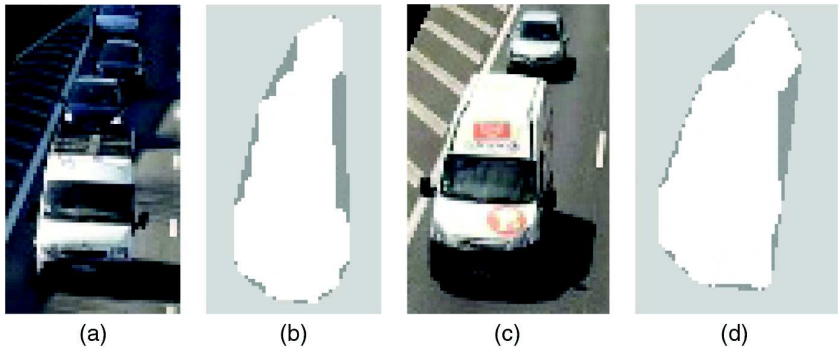

Fig. 12 Convexity study: (a) and (c) observed scene, (b) and (d) detected blobs (white) and their convex hull (dark gray).

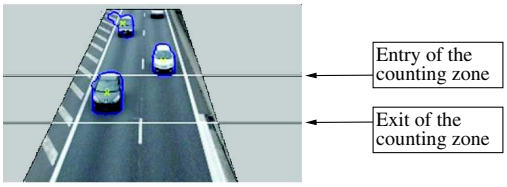

Fig. 14 Preliminary counting zone.

A vehicle is counted if it crosses the counting zone, i.e., if its trajectory begins before the entry line and continues after the exit line.

Then, vehicles are classified into three categories: light vehicles (LV: all traditional cars and small commercial vehicles, vans, and so on), heavy vehicles (HV: small and big trucks needing a different driving license), and two-wheelers (TW: motorbikes, mopeds). The classification is made according to their width compared to those of the road at the exit line level. As in our case, we are facing the road, and the width is a good discriminating indicator.

For some vehicles, like two-wheelers, the tracking begins later because of detection problems. In order to take into account these kind of vehicles, we add a second counting zone that overlaps the first one, as shown in Fig. 15. The second counting zone reinforces the counting procedure.

4 Results

In this section are detailed results of our shadow removal method and the entire counting system as well.

4.1 *Shadow Removal*

The shadow removal module has been evaluated on the Highway I video from the ATON project datasets[30] with the consent of the UCSD Computer Vision and Robotics Research Laboratory[31] in the Electrical and Computer Engineering Department at U.C. San Diego. ATON Highway I is a very interesting video sequence for shadow elimination. It contains many vehicles coming up in front of the camera. There are large shadows from moving vehicles and from the background. This video had been used in some articles for shadow removal.[18,32–34] Figure 16 illustrates one image of this sequence.

In order to perform a quantitative evaluation of our method and to compare it to a similar method, we have set up a ground truth composed of 64 frames in which we have manually segmented, on average, three areas corresponding to shadows (this ground truth can be requested from the authors). So, the total number of vehicles segmented is ∼200. The ATON Highway I video was used for that purpose. The performance of the proposed algorithms on shadow elimination is evaluated thanks to recall = number of detected true shadow pixels / number of true

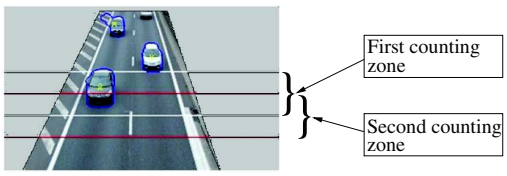

Fig. 15 Double counting zone.

Fig. 16 Image extracted from the Highway I video of the ATON project.

shadow pixels, and precision = number of detected true shadow pixels / number of detected shadow pixels. The numerous shadows carried by the vehicles present several configurations: vehicles far from the sensor, with small shadow areas, vehicles in the central part of the scene, and, finally, vehicles close to the sensor. Many difficulties appear on this setup. There are single vehicles but also on the same image, few vehicles merged by shadow.

Figure 17 shows the comparison of our method with Xiao's. In the first case, in which vehicles are isolated, for both methods, results are very similar most of the time, but our method performs much better in the second case, in which several vehicles are present in the scene. On average, from the 64 frames processed, our recall indicator is better than Xiao's (77 versus 62%). The precision scores are similar for the two methods.

Figure 18 shows the comparison between the two methods for only two images extracted from the ATON Highway I video. For the first one, we got a recall rate of 77.36 versus 42.13% for Xiao's method. For the second one, we obtained

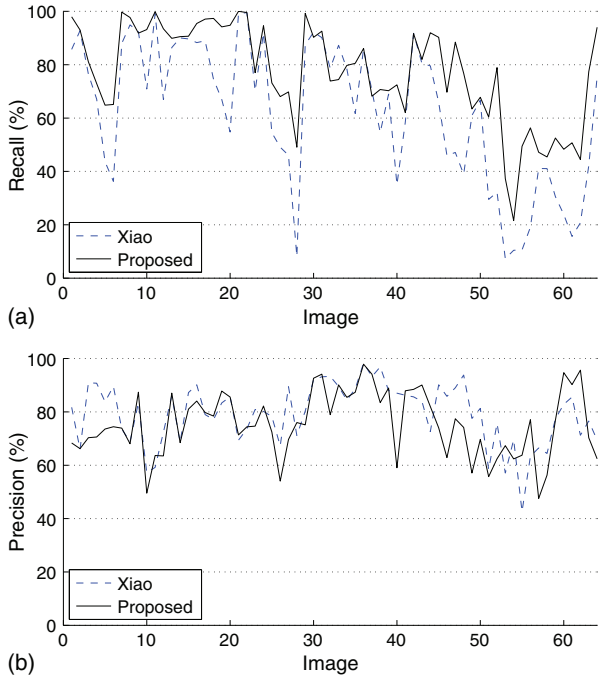

Fig. 17 (a) Recall and (b) precision comparison between our method and Xiao's (dashed line) on the 64 images setup.

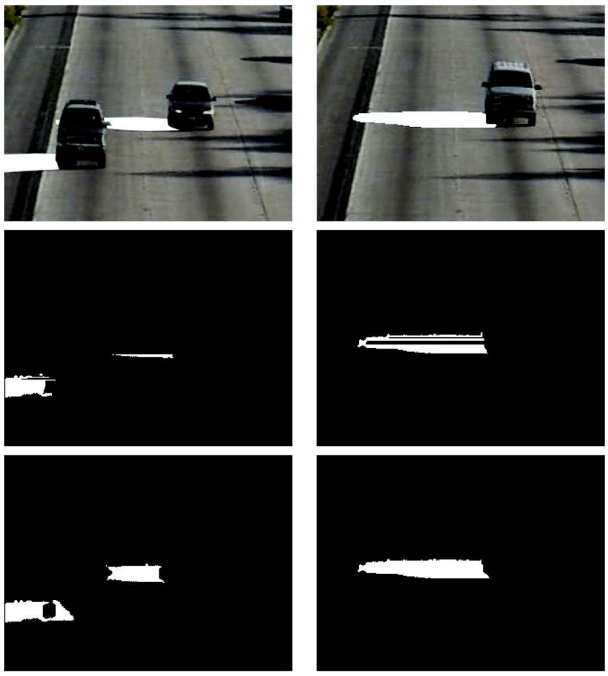

Fig. 18 Shadow removal comparison: results from (a) image no. 393 and (b) image no. 98. Raw images with vehicle shadows manually segmented on the first row. Vehicle shadows automatically segmented with Xiao method and from our moving region detection result on the second row. Vehicle shadows automatically segmented with our shadow removal module and from our moving region detection result on the last row.

94.15%, while Xiao's method achieves below 70%, 66.95% of recall rate.

4.2 *Occlusion Management*

In this section, we provide some figures on the evaluation of the occlusions. In principle, according to the position of the cameras (in the main axis of the road and relatively high), we try to avoid a maximum of occlusions. Nonetheless, according to the nature of the vehicles taken into account (trucks, cars, two-wheelers), there are, of course, some unavoidable occlusions. In these cases, we have analyzed them one by one and the software is able to handle more than 90% of the occlusion situations. Nevertheless, as noted in Sec. 3.4.2, there are some clinic situations for which it is quasi-impossible to deal with occlusions. These situations are a light vehicle hidden by a truck, a two-wheeled vehicle hidden by a light car or a truck, or situations in which the blobs resulting from vehicles are too overlapped to overcome the solidity test. In this case, only one final blob is considered, which leads to a subcounting. Another possibility to overcome this drawback is to install the video sensor higher, but it requires new installation features, which is not authorized by transport operators.

Fig. 20 Second-stage dataset with strong shadows.

4.3 *Vehicle Counting*

The evaluation work was divided into two stages. During the first stage, we acquired three different datasets on the same site (cf. Fig. 19). This site is also equipped with inductive loops, which are convenient for comparison purposes. The first dataset (named Cloudy) was shot during cloudy weather, and thus with cloudy illumination and without shadows. The second one (Sunny) was shot during a very sunny day and with severe shadows. The third one (Transitions) was shot in the presence of sparse clouds leading to sudden illumination changes. The three datasets are ∼20 min long and contain between 1300 and 1500 vehicles each, according to the ground truth.

During the second stage, a longer dataset was shot in another site and contains many difficulties due to shadows (cf. Fig. 20). It contains 3111 vehicles and is a 37-min-long video. Casted shadows from vehicles are more spread and stretched due to the sun position. In the observed scene, there are two kinds of shadows: those that are stationary and created by road panels, and those moving and coming from swaying branches. Moreover, as we are next at an exit road, the road marking is denser.

Table 1 shows the vehicle counting and classification results. The ground truth has been obtained manually. For each vehicle class, from the results automatically computed by our system, the number of false negatives (undetected vehicles), false positives (mistakenly counted vehicles), and misclassified (assigned to a wrong class) vehicles are calculated. The system is evaluated according to

- classification performance using recall = true positives / ground truth, and precision = true positives / (detected vehicles − misclassified); "total recall" and "total precision" are the averages of the values obtained with the three vehicle categories;
- detection performance using detection rate = 1 − false negatives / ground truth, false detection rate = false

Fig. 19 Dataset: (a) Cloudy, (b) Sunny with strong shadows, and (c) Transitions with abrupt illumination changes.

Table 1 Results of the proposed video-based vehicle counting system.

			Our system									Inductive loops	
			Results				Classification performance		Detection performance			Detection performance	
Dataset	Class	Ground truth	Detected vehicles	False negatives	Mis classified	False positives	Recall	Precision	Detection rate	False detection rate	Detection ratio	Detected vehicles	Detection ratio
Cloudy	LV	1425	1417	10	1	2	99.23%	99.86%					
	HV	46	46	1	1	1	95.65%	97.78%					
	TW	34	30	6	0	2	82.35%	93.33%					
	Total	**1505**	**1493**	**17**	**2**	**5**	**92.41%**	**96.99%**	**98.87%**	**0.33%**	**99.20%**	**1415**	**94.02%**
Sunny	LV	1196	1160	20	0	2	96.82%	99.83%					
	HV	67	81	4	14	4	94.03%	94.03%					
	TW	38	43	2	2	5	94.74%	87.80%					
	Total	**1301**	**1284**	**26**	**16**	**11**	**95.20%**	**93.89%**	**98.00%**	**0.85%**	**98.69%**	**1236**	**95.00%**
Transitions	LV	1284	1266	15	1	1	98.44%	99.92%					
	HV	19	20	3	4	1	78.95%	93.75%					
	TW	48	42	7	1	0	85.42%	100.00%					
	Total	**1351**	**1328**	**25**	**6**	**2**	**87.60%%**	**97.89%**	**98.15%**	**0.15%**	**98.30%**	**1266**	**93.71%**

positives / ground truth, and detection ratio = detected vehicles / ground truth.

The results obtained by inductive loops are evaluated using their detection ratio.

Based on the detection ratio, vehicle detection results of the video-based system are better than those obtained with the ILD system, whatever the dataset considered: 99.2 against 94.02% for Cloudy, 98.69 against 95% for Sunny, and 98.3 against 93.71% for Transitions. The detection rate of our system is always greater than 98% and the false positive rate is equal to 0.85% in the worst case. The detection results of two-wheeled vehicles are worse than those of light vehicles. The number of false negatives can be explained by several factors. Their small size, their high speed, and their nonrectilinear trajectories make them difficult to track with the Kalman filter. Moreover, the shadow removal process needs a minimum number of interior edges, which is rarely the case with two-wheelers. It can also be noted that the number of two-wheelers being low, the results have to be interpreted carefully. It is also the case for heavy vehicles in the Transitions dataset. The promising results on the first three datasets are confirmed by the counting results for the fourth dataset used in the second stage of our experiments. In spite of difficulties induced by road marking and shadows due to road panels, moving tree branches, and vehicles, the detection ratio remains very satisfactory, 98.33%, with 3059 vehicles detected out of 3111.

Finally, the results achieved here meet the transport operators' requirements, which are as follows: an acceptable result for them is an error of $\pm 3\%$ for 95% of the dataset processed. In our case, we obtained an error of around 2% on average for the entire database.

Regarding the classification performance, despite the simplicity of the criterion (width of the blob), results are rather satisfactory. For all the datasets, the best recall and precision are obtained for the LV category, with a mean recall equal to 98.16% and a mean precision of 99.87%. For the other vehicle categories, the number of misclassified vehicles comes from the low number of classes. Intermediate vehicles, like camping cars or vans for example, supposed to be classified into LV could be classified into heavy cars due to their width. It would be interesting to consider more classes of vehicles to reduce errors and to get a better characterization of the traffic — but taking into account that more categories would require a more sophisticated and discriminant classification criterion than the blob width.

The authors are ready to make the different datasets available for the community.

4.4 *Computation Time*

The global algorithm developed is able to process data at a 20 frames/s cadence. In our environment (ring roads), this represents a real-time functioning because with this processing time, every vehicle is taken into account whatever its speed. In a first stage, images were captured at a resolution of 720×576 pixels. Then, these images were converted into a smaller

resolution of 288×172 without affecting the counting accuracy. Of course, the different routines present different consumption with respect to the total. One can find below the main differences between these routines:

1. Motion detection: 45.48% (including GMM and updating, 31.07%, and moving region detection, 14.41%);
2. Tracking: 37.19%;
3. Shadow removal: 16.01%;
4. Occlusion management: 1.00%;
5. Trajectory counting: 0.31%.

5 Conclusion

In this work, we developed an advanced road vehicle counting system. The aim of such a system is to replace or complement in the future the old systems based on ILD. The system has been tested with different kinds of illumination changes (cloudy, sunny, transitions between sun and clouds), obtaining results better than those of ILD. The developed algorithm is able to eliminate several kinds of shadows depending on the time of the day. Another particular strength of the method proposed is its ability to deal with severe occlusions between vehicles. Multicore programming allows us to achieve real-time performances with only a piece of software. The perspective of this work is, with the same sensor, to continue to calculate traffic indicators like occupation rate or density of vehicles. The two previous indicators could be used to calculate more global congestion indicators. The infrastructure operators are very interested in having such statistics in real time for management purposes.

References

1. "Sensors for intelligent transport systems," 2014, http://www.transport-intelligent.net/english-sections/technologies-43/captors/??lang=en.
2. "Citilog website," 2015, http://www.citilog.com.
3. "FLIR Systems, Inc.," 2016, http://www.flir.com/traffic/.
4. S. Birchfield, W. Sarasua, and N. Kanhere, "Computer vision traffic sensor for fixed and pan-tilt-zoom cameras," Technical Report Highway IDEA Project 140, Transportation Research Board, Washington, DC (2010).
5. C. C. C. Pang, W. W. L. Lam, and N. H. C. Yung, "A method for vehicle count in the presence of multiple-vehicle occlusions in traffic images," *IEEE Trans. Intell. Transp. Syst.* **8**, 441–459 (2007).
6. M. Haag and H. H. Nagel, "Incremental recognition of traffic situations from video image sequences," *Image Vis. Comput.* **18**, 137–153 (2000).
7. L. Unzueta et al., "Adaptive multicue background subtraction for robust vehicle counting and classification," *IEEE Trans. Intell. Transp. Syst.* **13**, 527–540 (2012).
8. S. Greenhill, S. Venkatesh, and G. A. W. West, "Adaptive model for foreground extraction in adverse lighting conditions," *Lec. Notes Comput. Sci.* **3157**, 805–811 (2004).
9. K. Kim et al., "Real-time foreground-background segmentation using codebook model," *Real-Time Imaging* **11**, 172–185 (2005).
10. C. R. Wren et al., "Pfinder: real-time tracking of the human body," *IEEE Trans. Pattern Anal. Mach. Intell.* **19**, 780–785 (1997).
11. G. Gordon et al., "Background estimation and removal based on range and color," in *IEEE Computer Society Conf. on Computer Vision and Pattern Recognition*, pp. 459–464 (1999).
12. C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," in *IEEE Computer Society Conf. on Computer Vision and Pattern Recognition*, pp. 2246–2252 (1999).
13. C. Stauffer and W. E. L. Grimson, "Learning patterns of activity using real-time tracking," *IEEE Trans. Pattern Anal. Mach. Intell.* **22**, 747–757 (2000).
14. M. Harville, G. G. Gordon, and J. I. Woodfill, "Foreground segmentation using adaptive mixture models in color and depth," in *Proc. IEEE Workshop on Detection and Recognition of Events in Video*, pp. 3–11 (2001).
15. J. Rittscher et al., "A probabilistic background model for tracking," in *Proc. ECCV*, pp. 336–350 (2000).
16. B. Stenger et al., "Topology free hidden Markov models: application to background modeling," in *Proc. of Eighth IEEE Int. Conf. on Computer Vision*, pp. 294–301 (2001).
17. D. Grest, J.-M. Frahm, and R. Koch, "A color similarity measure for robust shadow removal in real time," in *Proc. of the Vision, Modeling and Visualization Conf.*, pp. 253–260 (2003).
18. A. J. Joshi et al., "Moving shadow detection with low- and mid-level reasoning," in *Proc. of IEEE Int. Conf. on Robotics and Automation*, pp. 4827–4832 (2007).
19. R. Avery et al., "Investigation into shadow removal from traffic images," *Transp. Res. Record: J. Transp. Res. Board* **2000**, 70–77 (2007).
20. X. F. Song and R. Nevatia, "Detection and tracking of moving vehicles in crowded scenes," in *IEEE Workshop on Motion and Video Computing*, pp. 4–8 (2007).
21. I. Huerta et al., "Detection and removal of chromatic moving shadows in surveillance scenarios," in *IEEE 12th Int. Conf. on Computer Vision*, pp. 1499–1506 (2009).
22. B. Johansson et al., "Combining shadow detection and simulation for estimation of vehicle size and position," *Pattern Recognit. Lett.* **30**, 751–759 (2009).
23. A. Sanin, C. Sanderson, and B. C. Lowell, "Shadow detection: a survey and comparative evaluation of recent methods," *Pattern Recognit.* **45**, 1684–1695 (2012).
24. N. Al-Najdawi et al., "A survey of cast shadow detection algorithms," *Pattern Recognit. Lett.* **33**, 752–764 (2012).
25. B. Coifman et al., "A real-time computer vision system for vehicle tracking and tracking surveillance," *Transp. Res. Part C* **6**, 271–288 (1998).
26. L. Z. M. Xiao and C.-Z. Han, "Moving shadow detection and removal for traffic sequences," *Int. J. Autom. Comput.* **4**(1), 38–46 (2007).
27. J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.* **PAMI-8**, 679–698 (1986).
28. G. Jun, J. K. Aggarwal, and M. Gokmen, "Tracking and segmentation of highway vehicles in cluttered and crowded scenes," in *Proc. of IEEE Workshop Applications of Computer Vision*, pp. 1–6 (2008).
29. J. Kittler and J. Illingworth, "Minimum error thresholding," *Pattern Recognit.* **19**, 41–47 (1986).
30. "Shadow detection and correction research area of the autonomous agents for on-scene networked incident management (ATON) project." http://cvrr.ucsd.edu/aton/shadow/
31. "UC San Diego Computer Vision and Robotics Research Laboratory." http://cvrr.ucsd.edu/.
32. A. Prati et al., "Detecting moving shadows: algorithms and evaluation," *IEEE Trans. Pattern Anal. Mach. Intell.* **25**, 918–923 (2003).
33. I. Mikic et al., "Moving shadow and object detection in traffic scenes," in *Proc. of 15th Int. Conf. on Pattern Recognition*, pp. 321–324 (2000).
34. M. M. Trivedi, I. Mikic, and G. T. Kogut, "Distributed video networks for incident detection and management," in *Proc. of 2000 IEEE Intelligent Transportation Systems*, pp. 155–160 (2000).

Alain Crouzil received his PhD degree in computer science from the Paul Sabatier University of Toulouse in 1997. He is currently an associate professor and a member of the Traitement et Comprehension d'Images group of Institut de Recherche en Informatique de Toulouse. His research interests concern stereo vision, shape from shading, camera calibration, image segmentation, change detection, and motion analysis.

Louahdi Khoudour received his PhD in computer science from the University of Lille, France, in 1996. He then obtained the leading research degree in 2006 from the University of Paris (Jussieu). He is currently a researcher in the field of traffic engineering and control. He is head of the ZELT (Experimental Zone and Traffic Laboratory of Toulouse) group at the French Ministry of Ecology and Transport.

Paul Valiere received his MS degree in computer science from the University Paul Sabatier at Toulouse, France, in 2012. He is currently a software engineer at Sopra Steria company.

Dung Nghy Truong Cong received her BS degree in telecommunications engineering from the Ho Chi Minh City University of Technology, Vietnam, in July 2004, her MS degree in signal, image, speech, and telecommunications from the Grenoble Institute of Technology (INPG), France, in July 2007, and her PhD in signal and image processing from the Pierre & Marie Curie University. She is currently an associate professor at Ho Chi Minh City University of Technology, Vietnam. Her research interests span the areas of signal and image processing, computer vision, and pattern recognition.