Vacant parking area estimation through background subtraction and transience map analysis

Carlos Gálvez del Postigo , Juan Torres, Jose-Manuel Menéndez

Abstract: A method for estimating the dimensions of non-delimited free parking areas by using a static surveillance camera is proposed. The proposed method is specially designed to tackle the main challenges of urban scenarios (multiple moving objects, outdoor illumination conditions and occlusions between vehicles) with no training. The core of this work is the temporal analysis of the video frames to detect the occupancy variation of the parking areas. Two techniques are combined: background subtraction using a mixture of Gaussians to detect and track vehicles and the creation of a transience map to detect the parking and leaving of vehicles. The authors demonstrate that the proposed method yields satisfactory estimates on three real scenarios while being a low computational cost solution that can be applied in any kind of parking area covered by a single camera.

1 Introduction

The use of smart parking management systems is increasingly gaining importance nowadays. The most of surveilled parking areas use individual sensors to monitor each parking lot, which is expensive and non-scalable. Furthermore, non-delimited areas cannot be monitored. Given the large number of vehicles in large cities and the progressive development of ITS, designing a scalable urban parking system becomes interesting: it can provide the users assistance to find the closest free parking area, saving time and money. Moreover, it can be used as a surveillance system for detecting not-allowed vehicles parked on restricted areas.

Many techniques have been proposed to address the parking lots classification problem using computer vision techniques. Some of them are based on single-image analysis; for example, Liu [1] proposed a QuadTree segmentation to detect vehicles. This technique gives false occupancy detection under the presence of shadows and other objects. Funck *et al.* [2] analysed the luminance of the parking lot to gain robustness against illumination conditions. Both techniques require empty parking lot reference images under certain conditions, so they may not be suitable for urban scenarios.

Most of the approaches require complex training stages. Huang *et al.* [3] proposed a surface-based inference framework for each parking lot, modelled as a three-dimensional cube previously learned. For each of them, histograms of oriented gradient features are extracted and incorporated to a probabilistic model to infer the optimal hypothesis of the parking status. They improved the model [4] with a Bayesian hierarchical framework to handle luminance variations, shadows, perspective distortion and occlusions between vehicles. More recently [5], they also included night-time operation. Other techniques use classifiers and manual selections of parking lots. Wu *et al.* [6, 7] used a multi-class support vector machine; Ichihashi *et al.* [8] used a fuzzy C-means classifier, improved by particle swarm optimisation. Although it addresses variable light conditions, occlusions may deteriorate the accuracy.

The other alternative, followed in this paper, is to take advantage of the motion analysis. Fujiyoshi and Kanade [9] showed a robust method to tackle occlusions between vehicles. Two layers are created: background and foreground, in order to distinguish stopped from moving vehicles. Chen *et al.* [10] combined vehicle tracking with colour and edge models to classify each parking state. However, they need multiple cameras to track vehicles along the entire parking area.

Contrary to [10], our method uses a single static camera that captures successive video frames of a parking area. It provides an estimation of the dimensions of the free parking regions. Our method works in not-delimited parking areas in outdoor urban scenarios. Therefore it faces complex scenarios involving the presence of pedestrians and other moving objects, variable light conditions and occlusions between vehicles.

Furthermore, our technique does not require a training stage, making its applicability wider. Our approach analyses the movement of the vehicles along the video frames and detects parking or leaving on a parking area. Tracking is performed to make the application robust against small moving objects and illumination changes.

Since our method uses the information between frames, it requires an initial analysis of the scene. This includes both a pre-selection of parking regions to be processed and an estimate of the parked vehicles in the first frame. In addition, we perform a geometric calibration to obtain real dimensions in metres.

The resulting algorithm has been tested under different conditions with satisfactory results, being robust against illumination changes, occlusions and shadows. The algorithm was tested under these conditions besides under raining. Compared to other algorithms, the proposed method has the advantages of using a single camera, running in real-time in a general-purpose computer and not requiring training. Contrary to most of the previous approaches, the algorithm is capable of estimating the dimensions of the parking slots making it adaptable to non-delimited parking areas. It is also robust against pedestrians, bicycles and other small objects, which often trigger false positives (FPs) in other approaches. We therefore contribute with a low-cost solution that offers reliable results and ease of use, therefore suitable for a quick deployment in urban scenarios. Although the obtained error rates are satisfactory from a research perspective, they might still be rather high for some real applications. Some further research is also proposed to reduce the error rates, thus it would lead to a more robust performance.

The rest of the paper is structured as follows: In Section 2, we describe the proposed method, which is later evaluated in Section 3. Finally, some conclusions and future work are posed in Section 5.

2 Method description

In the following sections, we describe the proposed method, whose structural description is presented in Fig. 1. After an initialisation, we perform background subtraction and create a transience map. From this map, vehicles are detected and tracked to determine their parking or leaving status and also account for occlusions. Later, an estimation of the vacant parking area is provided. Finally, we explain some recommendations to set up the parameters of the algorithm.

2.1 Initial analysis

The first configuration step consists of defining the parking regions to be analysed, from which a binary mask is created. Second, we process the first video frame establishing the estimation of the zones occupied by vehicles. For this purpose, we use a watershed algorithm [11] by making a rough selection of the parked vehicles. The obtained results are sufficient, since a security distance between parked vehicles exists: the estimation errors of the watershed algorithm are negligible compared to this.

Furthermore, we perform a geometrical calibration of the camera to obtain real-world measures of the parking lots dimensions. Many techniques can be applied to this problem [12]. We implement it by applying a four-point perspective transform, as proposed by Blinn [13], because minimal information of the scene is required and it can be applied in most of the real situations. In this way, we obtain the projection matrix that transforms every frame from the original perspective to that perpendicular to the road surface.

To obtain the right pixels-to-metres transform, we use the known distance in metres between two points of the image in pixels. Once the initial analysis is performed, we process each acquired frame using grey-scale information. Experiments with red–green–blue images were performed; the computational cost increased but the detection results did not improve significantly.

2.2 Background modelling

A background model is needed to detect the moving vehicles. For this purpose, we implement the Mixture of Gaussians technique, following the algorithm proposed by Kaew TraKul Pong and Bowden [14]. For the learning rate, we choose an intermediate value such that it classifies as background a moderate movement of trees, leaves and so on, but not vehicles and other moving objects at conventional urban velocities.

The background is also used to detect stopping objects (e.g. vehicles that are parked) by creating a transience map, which will be discussed in the following subsection.

2.3 Transience map

The transient map is a technique that performs well on detection of stopped vehicles, as studied by Kujanperä [15]. It is an 8-bit grey-scale map where each pixel is classified as:

(1) *Background (255):* Background of the scene.
(2) *Transient (127):* Moving object.
(3) *Stationary (0):* Previously moving object that recently stopped.

This method analyses the stability $S(x, y, t)$ at time t of a pixel (x, y) defined as the variance of the intensity over N_f future frames, according to (1)

$$S(t) = \frac{N_f \sum_{j=0}^{N_f} I(t+j)^2 - \left(\sum_{j=0}^{N_f} I(t+j)\right)^2}{N_f(N_f - 1)} \quad (1)$$

where $I(x, y, t)$ is the grey-scale intensity. In the rest of the paper, we avoid indexes x, y for simplicity on notation.

We define the value of each pixel $T(t)$ at time t on the transience map according to (2)

$$T(t) = \begin{cases} 127, & S > S_{th} \\ 255, & S < S_{th} \text{ and } |I(t) - B(t)| < B_{th} \\ 0, & S < S_{th} \text{ and } |I(t) - B(t)| > B_{th} \end{cases} \quad (2)$$

where $B(t)$ is the grey-scale intensity of the background model at time t; S_{th} and B_{th} are thresholds that determine whether a pixel belongs to a moving object or not, and whether it belongs to a new object compared to the current background, respectively. An example of the resulting transience map is shown in Fig. 2 for clarification.

From the transience map, we generate binary masks from stationary and transient pixels, which are then processed with morphological operations (dilation + erosion + opening) to remove noise and create homogeneous groups of pixels, also known as blobs.

2.4 Vehicle detection and tracking

From the transient and stationary masks, we analyse all the blobs to detect and track vehicles. This allows to precisely determine whether a vehicle has parked or not. Besides, it is key to keep a record of previous vehicle positions and process occlusions, as we discuss in the following sections.

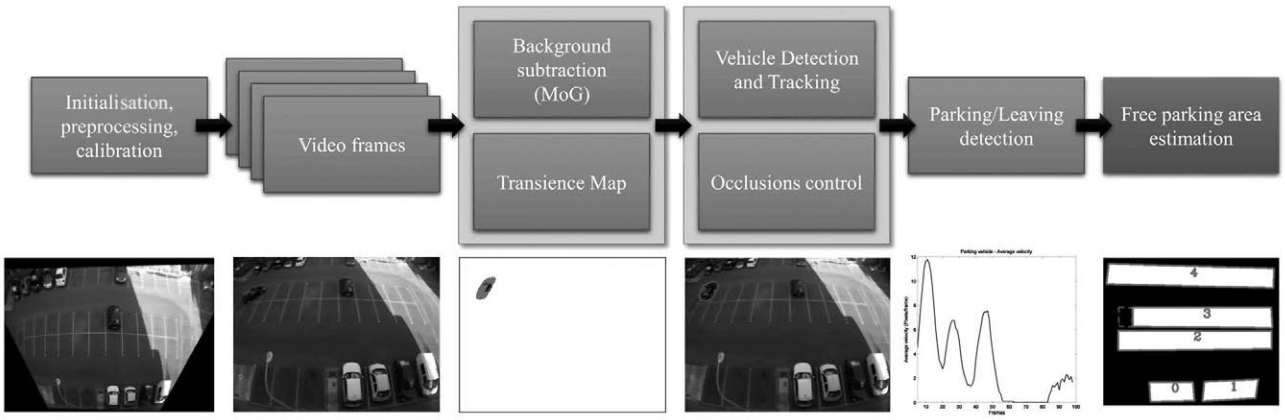


Fig. 1 *Block diagram of the proposed method*

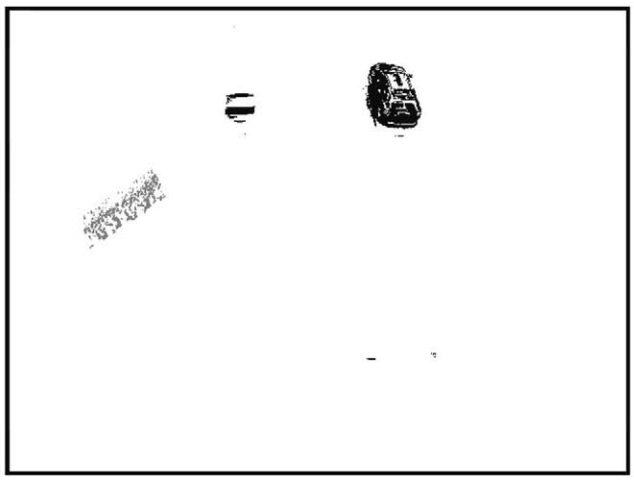


Fig. 2 *Transience map example*

On the top, two parked vehicles (the right-most is more recent)
The grey trace corresponds to a person walking

We define a vehicle model including the following features: mass centre (along N_v frames), and the visible and real contour. These can be easily extracted from the transient and stationary masks. We also define two structures, V_m, V_s, in which we store moving and stopped vehicles models, respectively. Moving vehicles models from V_m are analysed and updated according to the following algorithm.

First, we perform a matching between previously moving vehicles and current blobs. This is done by applying maximum likelihood estimation. We define the probability $P_i(t)$ that a blob with contour $C_i(t)$ at time t corresponds to the moving vehicle $v_i \in V_m$ detected on the previous frame, at $t-1$, with contour $C_i(t-1)$. We update the vehicle contour with $\hat{C}_i(t)$, which has maximum likelihood (whenever it is greater than a defined threshold)

$$P_i(t) = \frac{\|C(t) \bigcap C_i(t-1)\|}{\max\{\|C(t)\|, \|C_i(t-1)\|\}} \tag{3}$$

$$\hat{C}_i(t) = \arg\max_i \{P_i(t): P_i(t) \geq P_{th}\} \tag{4}$$

where the $\|C(t)\|$ operation computes the number of pixels contained within the closed contour $C(t)$. This guarantees a unique matching between new blobs and previous vehicles. This matching approach will properly work given that the frame rate is high enough in relation to the vehicle speed. In this case, the displacement of the vehicle from one frame to the next one will be small and there will be a sufficient intersection between the contours, therefore obtaining a high $P_i(t)$. In case a previous vehicle is not matched in the current frame, it is considered that is has left the scene.

The second step is to associate blobs to vehicles that may have started to leave a parking lot. We use the same procedure as before, but we match the current blob with the visible contour of the previously parked vehicle, $v \in V_s$.

Finally, the remaining blobs that have not yet been matched are analysed in terms of size and aspect ratio to decide whether they belong to a new incoming vehicle or not. For each remaining contour C_i, we create a new vehicle model out of it, $v_i(C_i)$ and store it in the moving vehicles structure, V_m, under the following conditions

$$V_m \leftarrow v_i(C_i) \Leftrightarrow \begin{cases} \|C_i\| & \in [S_{min}, S_{max}] \\ \mathrm{AR}(C_i) & \in [\mathrm{AR}_{min}, \mathrm{AR}_{max}] \end{cases} \tag{5}$$

where S_{min} and S_{max} determine the minimum and maximum size (in number of pixels) of the blob to be considered it belongs to a vehicle, and AR is the aspect ratio of the vehicle, to be within the range from AR_{min} to AR_{max}. The operation $V_m \leftarrow v_i(C_i)$ means that the vehicle v_i, generated from contour C_i, is stored in the data structure V_m.

This sequence is designed to match as much previous information as possible, and leave the processing of completely new blobs for the final steps. Otherwise, new vehicles would be detected on each frame and they would not be tracked.

Once we classify all blobs corresponding to vehicles, we decide whether changes have been produced in the parking lots.

2.5 Parking lots status decision

We analyse the two situations under which the parking areas status is modified: parking and leaving of vehicles, using the information stored in V_m and V_s.

2.5.1 Parking: The parking of each vehicle $v_i \in V_m$ is determined analysing its average velocity, $\bar{s}_i(t)$. It is approximated by a finite difference between the centroids at times t and $t-1$, and low-pass filtered by averaging over N_v past frames to reduce noise.

Fig. 3 represents an example of the velocity evolution of a parking manoeuvre. We distinguish three clear regions:

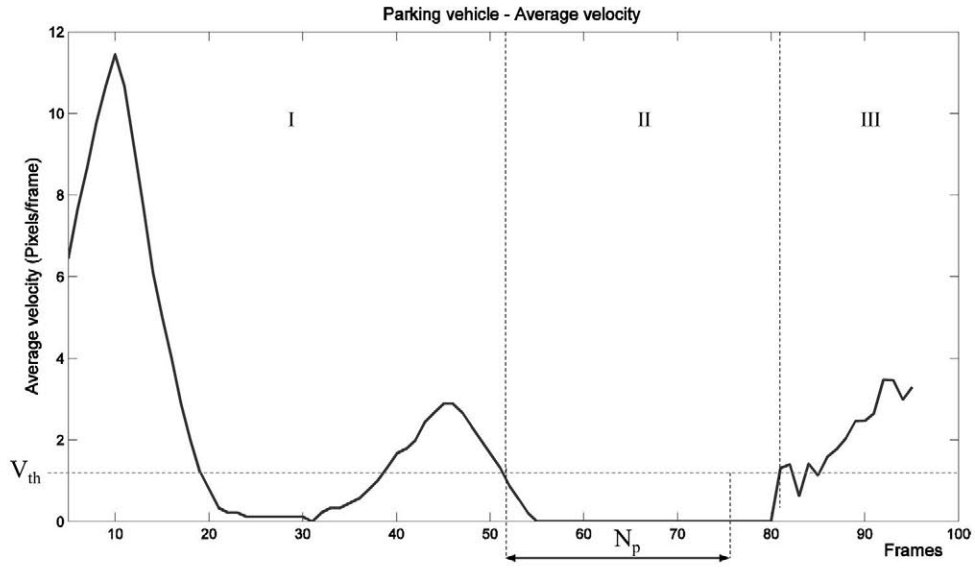


Fig. 3 *Average velocity of a parking vehicle*

Three clear regions (I, II, III) are observed

Fig. 4 *Transience map for a leaving vehicle*

I When the vehicle is moving around the scene, the velocity is high and variable.
II When the vehicle parks, the velocity decreases progressively down to a minimum value and is kept constant over a considerable number of frames.
III Once the vehicle has been parked for a while, it starts to belong to the background, so the associated blob starts to change in size and shape until it completely disappears, which produces changes in velocity.

Thus, the vehicles structures are updated as follows

$$\begin{matrix} V_{\mathrm{m}} \rightarrow v_i \\ V_{\mathrm{s}} \leftarrow v_i \end{matrix} \Leftrightarrow \bar{s}_i(t-j) < V_{\mathrm{th}}; \quad \forall j = 1, \ldots, N_{\mathrm{p}} \tag{6}$$

where $\bar{s}_i(t-j)$ is the average velocity of vehicle i at instant $t-j$ (where t is the current time), V_{th} establishes the maximum velocity of a vehicle to consider it as stopped, and N_{p} is the number of frames over which the velocity must be kept below this threshold (see Fig. 3). The operation $V_{\mathrm{m}} \rightarrow v_i$ removes the vehicle v_i from the data structure V_{m}. This temporal constraint is key to classify a vehicle as parked with a high level of confidence, reducing significantly the number of FPs. It also allows to ignore the manoeuvres when the velocity is below the threshold: intermediate positions are not stored, but the final parking position after several frames.

2.5.2 Leaving: When a vehicle leaves a parking lot, a stationary blob with the same shape and size appears in the place where the vehicle was before, as shown in Fig. 4. This is because it had previously been incorporated to the background.

When we detect such a blob, we try to match it with the already stored vehicle models $v \in V_{\mathrm{s}}$, following the same procedure as in Section 2.4. We also compare the size and aspect ratio to increase the confidence of the prediction and improve robustness. If the matching has been successful, the vehicle is removed from V_{s}, so it is not taken into account to further compute the occupancy of the parking area.

2.6 Occlusions handling

A common problem to be handled in our scenarios is the occlusions produced between vehicles, for which we use the tracking information. We consider occlusions produced between a parked vehicle, $v_{\mathrm{p}} \in V_{\mathrm{s}}$, and a vehicle that is going to park, $v_{\mathrm{m}} \in V_{\mathrm{m}}$:

1. v_{m} is going to park 'behind' v_{p} (Fig. 5). Here, we estimate the real contour of v_{m} once it has parked: otherwise, it would not be possible to determine the occupied area of v_{m} if v_{p} leaves in the future. Besides, we estimate the parked contour of v_{m} from the contour in the last frame prior to the occlusion, which is detected by intersecting the bounding boxes of v_{m} and v_{p}.
2. v_{m} is going to park 'in front of v_{p}' (Fig. 6). Here, we have the real contour and thus an estimation is not needed. Instead, we update the visible contour of v_{p} to detect the leaving of the parking lot (the size and the aspect ratio of the generated stationary blob is compared to the visible contour). To accomplish that, the new visible contour of v_{p} is obtained by subtracting the contour of v_{m} from the original one of v_{p}.

Lastly, we analyse the positions of the stopped vehicles in relation to the parking area and detect which ones are on parking lots.

2.7 Free parking lots measurements

From V_{s} and the projective matrix, we compute the effective free parking area available in metres.

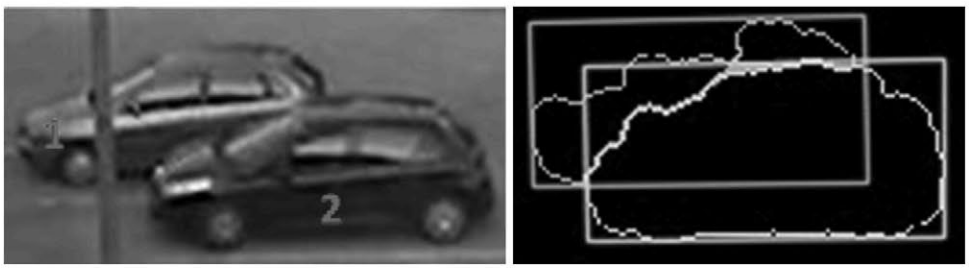
1. Given the mask of the total parking area, we compute its perspective transform (Section 2.1). Therefore real measures can be taken. We also transform the contour of each parked vehicle $v_i \in V_{\mathrm{s}}$.
2. The 'effective contour' of each vehicle is estimated and removed from the mask. It represents the occupied parking area, which may differ from the 'actual contour'. This becomes very useful in two aspects. First, subtracting this contour from the original mask yields a parallelogram, from which it is straightforward to compute distances. Second, this method accounts for bad-parked vehicles: if a vehicle is parked rotated, it will cover part of the adjacent parking lot, so the effective contour will be larger, as shown in Fig. 7.
3. The last step produces a modified mask of free parking areas, each one of them four-sized, from which we compute the width and length. Given a known distance, these dimensions are finally transformed to real distances, in metres. Thus, the real dimensions of each parking area are given.

Once our method has been described, in the following section we present the experiments that validate our proposal.

a *b*

Fig. 5 *Occlusions handling when vehicle parked behind* Q3

a In vehicle 2 parking is behind 1, so its real contour is not visible
b Approximated parked contour



a b

Fig. 6 *Occlusions handling when vehicle parked in front*

a Vehicle 2 parks in front of vehicle 1
b The visible contour of vehicle 1 is updated; the real contour is still known, as shown by the bounding box

2.8 Parameter settings

The proposed algorithm uses a set of user-defined parameters. Table 1 summarises the values or range of values used in the experiments. Some of them are dependent on the scenario, so we present some suggested ranges and guidelines on how to tune them.

The main tuning needs to be done on S_{th}, B_{th} and N_p. S_{th} was set to 120 in our experiments. The lower the threshold, the easier to detect moving vehicles, with the drawback of increasing the noise; large values will require a fast motion for vehicles to be detected. B_{th} was set to 16 in the experiments. Lower values allow the algorithm to detect a recently stopped object more easily, but again it increases the noise; larger values make the detection harder, gaining robustness against illumination changes, but reducing the detection of stopped vehicles. A good tuning of these two parameters should give a transience map similar to the one in Figs. 2 and 4. Finally, the parking time, N_p, determines the number of frames a vehicle needs to be stopped to mark it as 'parked'. A value of 30 frames was used in our experiments. Low values may produce too early classifications as 'parked'. Large values risk the parking detection since the vehicle might incorporate into the background before (6) holds (in clear region III according to Fig. 3). Finally, the values of S_{min}, S_{max}, AR_{min} or AR_{max} can be further adjusted in case the vehicles are too close or too far away from the camera, but it was not necessary in our experiments.

3 Experiments and results

To validate our algorithm, we acquired images from three parking scenarios using two different IP cameras. Images were acquired in colour with resolutions of 640 × 480 pixels (scenarios A and C) and 2048 × 1536 pixels (scenario B). They were coded in JPEG.

The proposed method was implemented in C++, taking advantage of the computer vision library OpenCV 2.4.6.0.

We used a general-purpose computer to run the application, with an Intel Core i7-2600 at 3.4 GHz, 8 GB DDR3 RAM. The application ran on Windows 7 SP1 (64 bits).

In the following subsection, we describe the video sequences used for testing our algorithm and explain the differences between the scenarios. Later, we demonstrate that our algorithm was capable to work in both scenarios, as shown by the detection rates over all the footage.

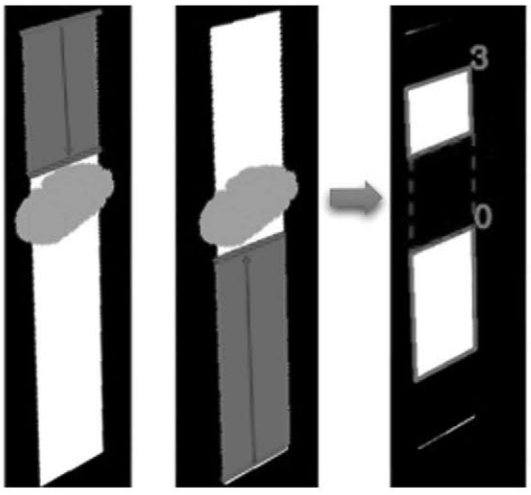


Fig. 7 *Effective contour estimation*

We create a bounding box around the vehicle contour by moving the shorter sides of the parking row towards the vehicle
The remaining free parking areas (labelled '0' and '3') are thus obtained

3.1 Scenarios description

The three scenarios are shown in Fig. 8. Cameras perspectives, illumination conditions and video quality were different in each scenario. In addition, shadows from buildings and trees, and occlusions of pedestrians and moving vehicles were issues to be handled in all of the scenarios.

For scenarios A and B, we analysed 3 and 4 h video sequences from each scenario, both in the morning and in the evening, corresponding to the highest flow of parking and leaving vehicles, respectively. For scenario C, 12 h video sequence was analysed. Rainy conditions are tested in sequence of 6 h in scenario A. Furthermore, owing to the low position of the Sun, shadows were prominent and shines were produced. In addition, pedestrians and bicycles often appeared throughout the footage, especially in scenario B. Therefore we also analysed how well the algorithm was capable of filtering them.

3.2 Results

We defined the following false negative (FN) rates and the number of FPs (NP) based on the detection of parking and leaving vehicles as a performance indicator

$$FN_p = \frac{N_{pn}}{N_{pT}} = \frac{\text{No. not detected parking vehicles}}{\text{No. total parking vehicles}}$$

Table 1 Parameter values

Parameter	Value
N_f	10 (frames)
S_{th}	100–150 (max: 255)
B_{th}	10–30 (max: 255)
N_v	10 (frames)
P_{th}	0.5
S_{min}	0.01% of image size ($H \times W$)
S_{max}	0.1% of image size ($H \times W$)
AR_{min}	1.5
AR_{max}	4
V_{th}	1.5 (pixels/frame)
N_p	10–50 (frames)

Fig. 8 *Sample images taken from each scenario*

$$\mathrm{FN_l} = \frac{N_{\mathrm{ln}}}{N_{\mathrm{lT}}} = \frac{\text{No. not detected leaving vehicles}}{\text{No. total leaving vehicles}}$$

$$N_{\mathrm{pp}} = \text{No. false detections of parking}$$

$$N_{\mathrm{lp}} = \text{No. false detections of leaving}$$

We considered that a vehicle is parked when it occupied a parking lot and it finally stopped. Not detecting it meant that its model was not added to V_s. We considered that a vehicle left when its parking lot was completely released. Its detection failed when the model was not removed from V_s. The FPs were produced when the system detected either the parking of a new vehicle or the leaving of one of the previously parked ones, while that event did not actually happen.

With the previous considerations, we analysed the footage and obtained the following results. First, the FN rates are presented in Table 2. The results for FPs are shown in Table 3. In this case, we evaluated the performance considering the number of FPs, the average number of parked vehicles in the scene, $\bar{N}_{\mathrm{p}}$, and the number of frames N_{frames}. Increasing N_{frames} or $\bar{N}_{\mathrm{p}}$ will increase the likelihood that a FP might be produced, given a fixed $\bar{N}_{\mathrm{p}}$ or N_{frames}, respectively.

Good rates were obtained in general for all of the scenarios, taking into account their complexity (light changes, occlusions, shadows, rain, pedestrians and bicycles and so on). Furthermore, considering the number of parked vehicles and the number of frames of the whole footage, the FPs were very few. The FN rate was usually higher, since it is more difficult to detect the parking of a vehicle than the leaving, because of a weaker foreground–background contrast in the former case.

The algorithm was robust against sudden illumination changes, thanks to the tracking analysis. In addition, 'static' or slow-moving shadows did not affect the detection of vehicles, which improved methods based on single-frame processing. Moreover, we observed that small moving objects like pedestrians, bicycles and so on, were correctly filtered because of a quite different size and aspect ratio as compared to cars. The only issue came when there was a big group of people walking together: it was detected as a vehicle. However, the blob shape was usually not consistent in time and the algorithm could not properly track it, so the condition in (6) was never fulfilled and new parked vehicle was not ever detected. A false leaving would not be detected either, since the group of people would have to be in the same place of the parked car (on top of it), which is physically impossible. Therefore they do not

Table 2 FN error rates

Scene	N_{pT}	$\mathrm{FN_p}$, %	N_{lT}	$\mathrm{FN_l}$, %
A-morning	15	14.3	0	—
A-evening	0	—	5	0
A-raining	1	0	8	12.5
B-morning	32	15.6	1	0
B-evening	2	0	27	7.4
C	4	20	15	26.7

When '–', no such case observed.

Table 3 FP error

Scene	N_{pp}	N_{lp}	$\bar{N}_p$	N_{frames}
A-morning	0	1	10.31	64 800
A-evening	1	0	3.21	86 400
A-raining	0	1	5.32	129 600
B-morning	1	1	28.35	43 200
B-evening	2	2	14.23	57 700
C	0	0	12.12	259 200

affect the parking lot characterisation, which makes our method suitable for urban scenarios.

Nevertheless, we observed that there were some sporadic errors (for both parking and leaving detection) produced by moving shadows (especially in the evening) or a high flow of vehicles that produced complex and almost complete occlusions. Vehicles with very similar colour to the road were difficult to detect as well, since the contrast with the background was not enough. The leaving detection was more robust since the difference in intensity between frames was usually very high. The error rate was higher for the parking situation because there were usually more manoeuvres and sometimes the vehicle was added to the background too early, before having really parked.

We also analysed the computational cost of the algorithm. Our implementation took 30–50 ms per frame (20–33 fps), for the 640×480 resolution, and 50–100 ms (10–20 fps) for 2048×1536. The proposed method therefore achieved a high frame rate that makes our algorithm capable to work with real-time constraints. The CPU load was about 10–15%, while it only required 110–150 MB RAM.

4 Discussion

We have presented an algorithm that is able to detect the occupancy of non-delimited parking areas by means of a single static surveillance camera. Current state-of-the-art methods require multiple cameras, complex training stages or computationally costly algorithms. In contrast, the proposed method uses a single camera, is able to run under real-time constraints and does not require any training, which usually makes more difficult the performance in complex urban scenarios. Therefore it can be easily deployed in any urban scenario at a low cost and impact.

The method is especially suitable to tackle the most common challenges in urban scenarios: illumination changes, occlusions, shadows, pedestrians and so on. Several experiments were performed under different conditions, varying illumination, camera position, density of vehicles and pedestrians and rainy conditions, and the results were satisfactory, with FN rates below 27% and a maximum of 2 FP (on parking and leaving detection). The algorithm correctly filtered spurious detections such as pedestrians and sudden light changes, which often occur in urban scenarios. Nevertheless, the proposed method has limitations under poor illumination conditions, especially at night since the vehicles' headlights generate incorrect moving blobs in the transience map.

5 Conclusions

The proposed method yields good detection error rates taking into account the complex conditions of real urban scenarios regarding outdoors lights and occlusions with moving objects and other vehicles. It is also capable of working on real time. Besides, neither a previous knowledge of the parking lots nor a training stage is necessary. Thus, it is easy to be deployed in a real environment and does not require interventions in the asphalt, as other technologies do.

In addition, our method is flexible, since it is not restricted to delimited parking lots, but any type of parking area. Thus, it may be easily extended to tackle with enforcement purposes such as stopped vehicles in not-allowed areas. Even though it was tested in outdoor parking areas only, we expect that it would also perform well in roofed parkings, as long as the position of the camera avoids large occlusions between vehicles. Owing to better and less-variant illumination conditions, we expect to have lower error rates and more accuracy in these environments. Besides, the algorithm is easy to configure, which allows for a quick deployment in many different scenarios.

The proposed method was also tested under difficult illumination conditions. It was robust against slow-moving shadows and sudden illumination changes. It did not perform well in dark scenes (at night) because of poor illumination conditions. It was also tested under rainy conditions, and low errors were obtained. Nevertheless, some limitations are expected under poor visibility conditions, such as the presence of mist or snow.

The experiments show a rather high FN rate (especially in test scenario C), which might make the proposed work not robust enough for estimating the real number of parked vehicles in a real environment. Nevertheless, the possible applications and the results indicates that the use of computer vision techniques for parking lot detection is a promising field of research. Thus, we propose several approaches for future research which would lead to improvements on the FN rate. First, using the colour information (and different colour spaces) instead of a single grey-scale image could improve the background segmentation in the generalised method of moment procedure. In addition, the detection of the parking event could be more robust by estimating the vehicle's state over time using a hidden Markov model, which would detect the event automatically instead of directly relying on the fixed parking time N_p. Moreover, our current work focuses on combining our technique with single-image analysis to periodically perform automatic error corrections to improve robustness and achieve lower error rates. A parallel algorithm (e.g. based on texture analysis) would then analyse single frame and determine an estimate of the occupancy. Since it would not be capable of running in real-time, it would be called with a given periodicity, depending on the complexity of the scenario (e.g. every hour).

6 References

1 Liu, S.: 'Robust vehicle detection from parking lot images'. Technical Report no. ECE-2005-06, Boston University, Department of Electrical and Computer Engineering, 2005, pp. 10–20

2 Funck, S., Mohler, N., Oertel, W.: 'Determining car-park occupancy from single images'. Intelligent Vehicles Symposium, 2004 IEEE, 2004, pp. 325–328

3 Huang, C.-C., Dai, Y.-S., Wang, S.-J.: 'A surface-based vacant space detection for an intelligent parking lot'. 2012 12th Int. Conf. on ITS Telecommunications (ITST), 2012, pp. 284–288

4 Huang, C.-C., Wang, S.-J.: 'A hierarchical Bayesian generation framework for vacant parking space detection', *IEEE Trans. Circuits Syst. Video Technol.*, 2010, **20**, (12), pp. 1770–1785

5 Huang, C.-C., Tai, Y.-S., Wang, S.-J.: 'Vacant parking space detection based on plane-based Bayesian hierarchical framework', *IEEE Trans. Circuits Syst. Video Technol.*, 2013, **23**, (9), pp. 1598–1610

6 Wu, Q., Huang, C., Wang, S.-Y., Chiu, W.-C., Chen, T.: 'Robust parking space detection considering inter-space correlation'. 2007 IEEE Int. Conf. on Multimedia and Expo, 2007, pp. 659–662

7 Wu, Q., Zhang, Y.: 'Parking lots space detection' (Carnellie Mellon University, 2013)

8 Ichihashi, H., Notsu, A., Honda, K., Katada, T., Fujiyoshi, M.: 'Vacant parking space detector for outdoor parking lot by using surveillance camera and FCM classifier'. IEEE Int. Conf. on Fuzzy Systems, FUZZ-IEEE 2009, 2009, pp. 127–134

9 Fujiyoshi, H., Kanade, T.: 'Layered detection for multiple overlapping objects'. Proc. 16th Int. Conf. on Pattern Recognition, 2002, vol. 4, pp. 156–161

10 Chen, L.-C., Hsieh, J.-W., Lai, W.-R., Wu, C.-X., Chen, S.-Y.: 'Vision-based vehicle surveillance and parking lot management using multiple cameras'. Sixth Int. Conf. on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP), 2010, pp. 631–634

11 Meyer, F.: 'Color image segmentation'. Int. Conf. on Image Processing and its Applications, 1992, pp. 303–306

12 Kanhere, N., Birchfield, S.: 'A taxonomy and analysis of camera calibration methods for traffic monitoring applications', *IEEE Trans. Intell. Transp. Syst.*, 2010, **11**, (2), pp. 441–452

13 Blinn, J.F.: 'Inferring transforms', *IEEE Comput. Graph. Appl.*, 1999, **19**, (3), pp. 93–98

14 Kaew TraKul Pong, P., Bowden, R.: 'An improved adaptive background mixture model for real-time tracking with shadow detection'. Video-Based Surveillance Systems, Springer, 2002, pp. 135–144

15 Kujanperä, V.: 'Highway video surveillance–development of an automatic detection system for abnormal incidents employing computer vision'. Master's Thesis, University of Oulu, Department of Electrical and Information Engineering, 2010