

Article

A Real-Time Automatic Plate Recognition System Based on Optical Character Recognition and Wireless Sensor Networks for ITS

Nicole do Vale Dalarmelina ^{*}, Marcio Andrey Teixeira  and Rodolfo I. Meneguette 

Department of Informatics, Federal Institute of São Paulo, Catanduva 15808-305, Brazil;
marcio.andrey@ifsp.edu.br (M.A.T.); meneguette@ifsp.edu.br (R.I.M.)

^{*} Correspondence: nicole.dalarmelina@aluno.ifsp.edu.br

Received: 1 October 2019; Accepted: 15 December 2019; Published: 20 December 2019



Abstract: Automatic License Plate Recognition has been a recurrent research topic due to the increasing number of cameras available in cities, where most of them, if not all, are connected to the Internet. The video traffic generated by the cameras can be analyzed to provide useful insights for the transportation segment. This paper presents the development of an intelligent vehicle identification system based on optical character recognition (OCR) method to be used on intelligent transportation systems. The proposed system makes use of an intelligent parking system named Smart Parking Service (SPANS), which is used to manage public or private spaces. Using computer vision techniques, the SPANS system is used to detect if the parking slots are available or not. The proposed system makes use of SPANS framework to capture images of the parking spaces and identifies the license plate number of the vehicles that are moving around the parking as well as parked in the parking slots. The recognition of the license plate is made in real-time, and the performance of the proposed system is evaluated in real-time.

Keywords: intelligent transportation system; automatic license plate recognition; wireless sensors networks; optical character recognition

1. Introduction

The number of motor vehicles is growing in most places of the world at an alarming rate, causing big issues like traffic congestion, low quality of air and so on. According to McKinsey [1], it is expected that the number of vehicles will increase to 1.32 billion units by 2020. On the other hand, the development of Intelligent Transportation Systems (ITSs) is becoming a feasible solution for many current issues. ITS is the junction of various technologies that focus on service and application delivery that will monitor and manage the transportation system, making it more comfortable and secure [2].

Wireless Sensor Network (WSN) is one example of technology that has been used to build ITS solutions. WSN has received considerable attention from researchers, governments and businesses due to its potential to provide easy and cost-effective solutions in many different areas [3]. Moreover, WSN can be integrated with other technologies to assist ITS services [4]. Automatic License Plate Recognition (ALPR) is an example of ITS service that can be built using WSN technologies.

ALPR has become a very important tool in the ITS field because this service helps in the monitoring and control of the vehicles. However, due to the diversity of plate formats (for example plate size, plate background, character size, plate texture and so on), the accurate development of an ALPR system is a challenging task, mainly when it comes to an open environment, where there are variations in the illumination conditions during image capturing [5].

According to Patel et al. [6], the development of a conventional ALPR system can be divided into four steps: (1) vehicle capture image, (2) number plate detection, (3) character segmentation,

and character recognition. The main functions of steps (1) and (2) are to detect the object (vehicle) and capture the image to find out the potential regions within the image that may contain the license plate. The discussion of the object detection algorithms is beyond the scope of this paper. However, we refer the reader to notable works in [7,8] for detailed technical discussions of these algorithms. Regarding step (3), the main function is to isolate the foreground characters from the background within the detected license plate region. In step (4), methods are employed to try recognize the characters.

The ALPR system can have two variations: on-line and off-line. As described in [9], in the online ALPR system, the localization and interpretation of license plates take place instantaneously from the incoming video frames, enabling real-time tracking of moving vehicles through the surveillance camera. On the other hand, an offline ALPR system captures the vehicle images and stores them in a centralized data server for further processing.

Based on the characteristics above described, several methods have been proposed for ALPR systems, where each one has its advantages and disadvantages [10,11]. Most works of ALPR system [12,13], use the edge proprieties as features for localizing the license plate regions, and some of these works [14] capture the image of a vehicle carefully placed in front of the camera taking a clear image of the license plate. However, in a practical scenario, there may be huge variations in lighting conditions that make plate recognition even more difficult. Even having several works about the ALPR system, the development of an ALPR system that can work out accurately on open environments with variations on the illuminations is still necessary.

An intelligent parking service based on wireless sensors for managing public or private spaces named Smart Parking Service (SPANS) has been developed by [15]. The SPANS system is a framework built using low-cost sensors coupled in a node to monitor the parking spaces. The nodes capture the images of the parking spaces and periodically send them to the intelligent transport system. The system provides mobile access in which the user can perform various management functions, such as finding vacant parking lots and a statistical report.

In this paper, a vehicular identification system based on OCR and WSN for ITS is proposed. The proposed system is implemented as a new functionality inside of the SPANS system described in [15]. The system has access to the images from the camera and uses an OCR to detect the license plate number of the vehicles that are moving around the parking as well as parked in the parking slots. Once a new vehicle is detected, the proposed system detects the region of the plate and applies an algorithm to extract the image of the plate. This image is processed and an OCR method is used to extract the characters and numbers that compose the license plate. This information is stored on a cloud and it can be used, for example, in the department of transportation to detect stolen vehicles or issues in the vehicle's documentation. The performance of the proposed algorithm to recognize the license plate characters is analyzed in an open environment where the SPAN system is installed. So, the experiments are carried out in a real environment in real-time. The main contribution of the paper is the development of the ALPR system to be used in an open environment as well as the evaluation of the Tesseract 4.0 in real time in an open environment. The proposed system is evaluated in a real environment in real-time.

The remainder of the paper is organized as follows: Section 2 presents a brief background of the SPANS system, the OCR, and the related works. Section 3 describes the proposed system. Section 4 presents the scenario and the performance measurements. Section 5 discusses our results. Finally, Section 6 concludes the paper with a summary of the main points and outcomes.

2. Background

In this section, we briefly present a description of the SPANS Parking Service used as our framework to insert the proposed system as well as the related works.

2.1. The SPANS Parking Service Framework

The SPANS is a framework used to detect available parking slots. This service is integrated with an intelligent transportation architecture that provides the structure necessary to detect, manage and notify the vehicles about the space. The SPANS infrastructure uses sensors to detect parking slots and cameras to record activities around the parking spaces. Sensors and cameras are used to detect significant parking information so that it is possible to verify if there are any spaces available. The infrastructure consists of a data center (cloud) which provides mechanisms for data abstraction and data processing, as well as communication mechanisms between drivers and sensors. Figure 1 shows an abstraction of the SPANS infrastructure that is used to detect and notify users about free parking.

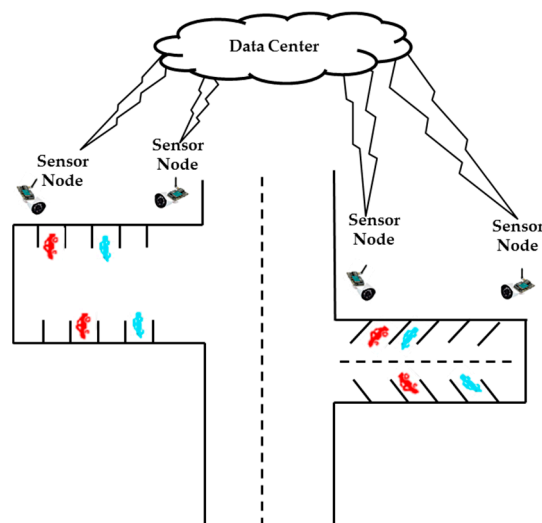


Figure 1. The infrastructure used by SPANS [5].

As can be seen in Figure 1, the data center has sensors that detect changes in the environment and receive information about the parking slots. This information is processed by the SPANS that will detect, using computer vision techniques [16], whether the parking slots are available or not. After the image processing, the system will provide information for the users about the available slots through a driver application.

2.1.1. The Data Center

The data center receives information (images) from sensors and sends information to mobile applications. This is done through a web RESTful (Representational state transfer) service which is implemented in the data center. So, images are captured by the sensors, processed, and the information about the availability of the parking slots are sent to the mobile applications. All data collected by the sensors are processed, analyzed and stored in a database that is accessible from the mobile application.

The wireless sensor node is based on a Raspberry Pi [17] model B. Raspberry Pi has a USB wireless network interface and a traditional webcam. Raspberry uses the Raspbian operating system, a Python program to read data from the GPIO (General Purpose Input/Output) pins and a program to monitor the webcam video signals. When a movement is detected, the web can wait a few seconds and take a picture of the vehicle that is arriving or leaving the parking slot.

2.1.2. The Parking Slots Information

The parking area is divided into several spaces (slots), as shown in Figure 2. Originally, each parking space has the following information:

- Park_Id: Parking space identification;

- Desc_park: Description of the parking space;
- X_coor: Coordinate “X” of the parking space;
- Y_coor: “Y” coordinate of parking space in parking areas;
- Width: Width of parking space;
- Height: Height of the parking space;
- Status: Stores the current status of the parking space. 0 is available; 1 is busy;
- Plate_N: Stores the plate license information.

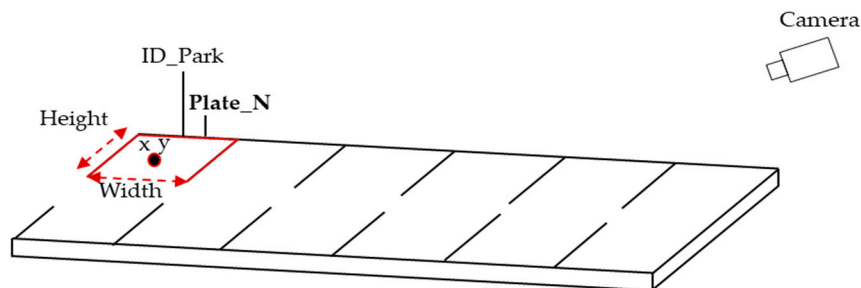


Figure 2. Parking information.

In this work, we inserted new information named “Plate_N” in the SPANS framework. So now, the information detected by our system is stored in the SPAN system and all information about the parking is sent to a database. Once the parking spaces have been defined, using a camera, the system receives an image called “Base Image” of each parking space, taking into account the X and Y coordinates and the width and height values. The “Base Image” is stored in the database and it is used to check if the parking space is available or occupied. The details about the SPAN service framework can be found in [15].

2.2. Optical Character Recognition

Optical Character Recognition (OCR) is one of the most widely studied problems in the field of pattern recognition and computer vision [18]. Despite being widely studied, OCR remains a challenging problem when used in unconstrained environments like parking areas. Moreover, it is necessary for improving the recognition rate on ambiguous characters, such as (B-8), (O-0), (I-1), (A-4), (C-G), (D-O), (K-X), and broken characters [19]. Several studies about the performance of the OCR are made using a dataset of images. In this paper, the OCR used is the Tesseract 4.0 [20]. The version 4.0 of the Tesseract implements a Long Short-Term Memory (LSTM)-based recognition engine which is a kind of Recurrent Neural Network (RNN). Its performance to recognize characters is made in real-time, where the images collected from the cameras are sent to the data center, processed and then used as input to Tesseract that will detect the license plate characters.

2.3. Related Works

The number of vehicles has been increasing in the last few years, and the ALPR has become a very important tool that helps in the monitoring and control of the vehicles. However, the existing diversity of plate formats, different scales, rotations, and no uniform illumination conditions during capturing the image [5] have become a big issue for the ALPR system. There are several methods and researches about ALPR. However, most of them are not applied in real-time. In this subsection, the related works are highlighted.

In [21], a project whose main objective is the improvement of the digits and features representations through Deep Learning techniques is presented. The authors developed a dataset composed of approximately 2000 images that are divided into a training set and test set. The authors conclude that the data augmentation techniques and the increase in locally connected layers have greatly increased the accuracy obtained with the datasets used in their approach.

In [22], the authors introduced a character recognition algorithm based on supervised classification. The authors make a description of the character classes using model the pixel sequence behavior in texts present in images. To reach the results, the strategy used is to describe the characters framed in a mesh, observing the transitions between the levels of gray from pixels, therewith, the pixel behavior in each class was determined, what could be used to classify other records. The authors concluded that the proposed algorithm in their project has obtained reasonable results for tasks that require real-time digit identification even with handwriting characters, recognizing 92,336 characters per minute.

In [23], a system was presented for the recognition of numerical digits in license plates. For the position recognition of the license plate, the White Top-Hat Transform was used so that the bright areas over dark objects could be removed and the image brightness could be corrected, soon after, the binarization process was done to transform it in black and white. After the localization of the license plate, a character's segmentation in a horizontal and vertical projection of the plate was made and the numbers were transformed into vectors. Finally, a multilayer perceptron network was resorted to for character recognition. According to [23], the classifier identified 37 out of 43, achieving an accuracy of 76%.

In [24], an approach was proposed to detect license plates in varying illumination conditions. A binarization method was used on the pre-processing step for plate segmentation, and the thresholding method was applied to the image. However, the experiment environment and processing time were not mentioned in the paper. In [25], a pre-processing and segmentation method for number plate localization was used. In the pre-processing, a global threshold was used to map the color intensity into a grayscale. The authors assumed two points: plates are oriented horizontally and there is a significant intensity difference between plate background and character foreground. However, in a real environment, these assumptions can change.

3. The Propose Vehicular Identification System Based on OCR for ITS

In this section, we introduce the algorithm developed to make the plate license recognition.

3.1. System Overview

The proposed system aims to read frames of a camera and recognize the characters present on license plates in real-time. Figure 3 shows the flowchart of the proposed algorithm.

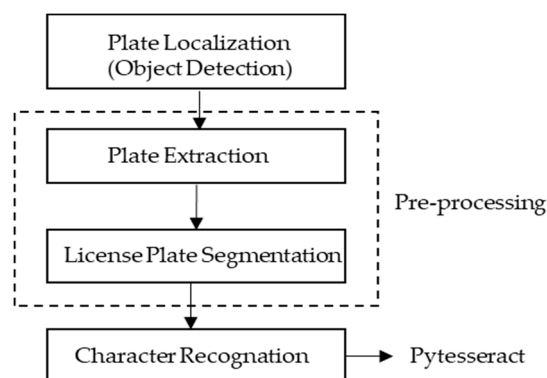


Figure 3. Flowchart of the proposed algorithm.

When SPAN detects a vehicle, the proposed algorithm starts the plate localization function to extract the license plate from the vehicle image. This is made by recognizing rectangular objects on the vehicle image. When the proposed algorithm finds a plate on the vehicle image, it extracts the plate area creating a new image. Once the plate image is extracted, it starts the pre-processing phase of the image. The pre-processing is necessary due to variations of the environment (illumination, background, texture) and is aimed at reducing possible noises on the image. It is used in the proposed system as a bilateral filter [26] to smooth the image and reduce the noise. After the pre-processing

phase, the proposed algorithm uses the Tesseract [20] to make the recognition of the characters present on the plate image. The pseudo-code of the proposed algorithm is shown in Section 3.2 below.

3.2. The Algorithm

In this subsection, how the proposed algorithm works is described in Algorithm 1, since the plate image detection until the character recognition. The proposed algorithm is described below, and Figure 4 [27] shows the results of the proposed algorithm using the images taken from the sensors.

Algorithm 1: The proposed algorithm

```

1. Start Video Capture
2. Read Video_Frame
3. Resize Video_Frame
4. Gray_Frame = Transform to grayscale (Video_Frame)
5. Apply filter to remove the noise (Gray_Frame)
6. Frame_Edges = Detect edges (Gray_Frame)
7. Frame_Contours = Find contours (Frame_Edges)
8. Sort (Frame_Contours)
9. Declare Number_Plate_Contour
10. Declare Largest_Rectangle
11. for Contour in Frame_Contours do
12.     Perimeter_Rectangle = Calculate perimeter (Contour)
13.     Approx_Rectangle = Find the approximate rectangle (Perimeter_Rectangle)
14.     if (length (Approx_Rectangle) == 4) then
15.         Number_Plate_Contour = Approx_Rectangle
16.         Largest_Rectangle = Find area (Approx_Rectangle)
17.     break
18.  $x,y,w,h$  = Calculate up-right bounding rectangle (Largest_Rectangle)
19. Cropped_Frame = Crop using  $x,y,w,h$  (Video_Frame)
20. Draw Largest_Rectangle contours on Video_Frame
21. Transform to grayscale (Cropped_Frame)
22. Frame_Threshold = Binarize (Cropped_Frame)
23. Kernel = New square object of size 1x1
24. Image_Dilation = Dilates using Kernel (Frame_Threshold)
25. Dilated_Image_Contours = Find contours (Image_Dilation)
26. Sorted_Dilated_Contours = Sort (Dilated_Image_Contours)
27. for Dilated_Contour in Sorted_Dilated_Contours
28.      $x,y,w,h$  = Calculate up-right bounding rectangle (Dilated_Contour)
29.     Draw a rectangle of dimensions  $x,y,w,h$  on Video_Frame
30. end for
31. Transform to binary (Gray_Frame)
32. License_Plate_Characters = Transform to string (Gray_Frame)
33. if length(License_Plate_Characters) > 0 then
34.     Get License_Plate_Characters
35. end if
36. return Video_Frame

```

As can be seen in the proposed algorithm, in the first line the video camera is started. So, the video camera reads and resizes the frames (lines 2, 3). The frames are resized following the distance of the camera to the vehicle to have a better perspective of the plate. From line 4 to line 7, filters are used to find the contours of the image. For this purpose, firstly a grayscale filter is applied in the image as illustrated in Figure 4b. After that, a bilateral filter is applied to remove noises on the image as shown in Figure 4c. So, the noises could be removed without damaging the edges of the frame, and the edges are detected as shown in Figure 4d. A loop starts at line 11 to find the rectangle that has the plate

numbers. Once found, this rectangle is cropped as illustrated in Figure 4e (line 19 of the algorithm). The block that includes the lines 21 to 31 exposes the pre-processing of the cropped image of the main frame. Figure 4f represents the result of this preprocessing. Finally, the characters are read at line 34 of the algorithm.



Figure 4. The proposed algorithm applied to the image to detect the characters. (a) Original Image, (b) Grayscale conversion, (c) Bilateral filter applied, (d) Edges detection, (e) The license plate cropped from the main frame, (f) The license plate after the treatment.

4. Evaluation Scenario and Performance Measurements

The proposed system is implemented in Python. The hardware used is an Alienware, Intel Core I7, 16 GB of RAM, 1 TB of HD. The Tesseract version used is the 4.0 [20]. The proposed algorithm is evaluated in real-time using real parking. Figure 5 shows the evaluation scenario, where the red dotted lines show the vision angle of the camera regarding the parking slots. Every time a car arrives, the camera takes a picture of the vehicle and sends the picture to the Data Center that runs the proposed algorithm to find the plate characters as described in the proposed algorithm in Section 3.2.

The performance of the proposed algorithm is measured by metrics derived from the confusion matrix [28,29]. The recognition of a license plate can only be considered correct if all characters that compose the license plate are correctly recognized. There are some cases where the license plate is incorrectly recognized (one or more characters are incorrectly recognized), or it is not recognized (any character is recognized). In this case, we use the generalized confusion matrix for multiple classes [30] to verify the performance of the proposed solution, where its class corresponds to a character. Table 1 shows the confusion matrix with n classes.

Table 1. Confusion matrix with n classes.

		Predicted Character					
		Class 1	Class 2	Class 3	Class 4	...	Class n
Actual Character	Class 1	x_{11}	x_{12}	x_{13}	x_{14}	...	x_{1n}
	Class 2	x_{21}	x_{22}	x_{23}	x_{24}	...	x_{2n}
	Class 3	x_{31}	x_{32}	x_{33}	x_{34}	...	x_{3n}
	Class 4	x_{41}	x_{42}	x_{43}	x_{44}	...	x_{4n}

	Class n	x_{n1}	x_{n2}	x_{n3}	x_{n4}	...	x_{nn}

Each class showed in Table 1 corresponds to a character used to compose the license plate. In this work, the following characters: "0", "1", "2", "3", "4", "5", "6", "7", "8", "9", "A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K", "L", "M", "N", "O", "P", "Q", "R", "S", "T", "U", "V", "X", "W", "Y", "Z" are considered. Considering the confusion matrix with n classes showed in Table 1, the total number of True Negative (TN), True Positive (TP), False Positive (FP) and False Negative (FN) for each class n are calculated as follows:

$$Total_FN_i = \sum_{\substack{J=1 \\ J \neq i}}^n x_{ij} \quad (1)$$

$$Total_FP_i = \sum_{\substack{J=1 \\ J \neq i}}^n x_{ji} \quad (2)$$

$$Total_TN_i = \sum_{\substack{J=1 \\ J \neq i}}^n \times \sum_{\substack{K=1 \\ K \neq i}}^n x_{jk} \quad (3)$$

$$Total_TP_{all} = \sum_{J=1}^n x_{jj} \quad (4)$$

Considering the generalized confusion matrix, the overall accuracy is calculated in accordance to Formula (5):

$$Overall\ Accuracy = \frac{Total_TP_{all}}{Total\ Number\ of\ Testing\ Entries} \quad (5)$$

The processing time measurement represents the time spent processing the image to recognize the characters and store the plate numbers in the database, where this information is available to be accessed by the web application. Moreover, the processing time is calculated taking into account the environment illumination.

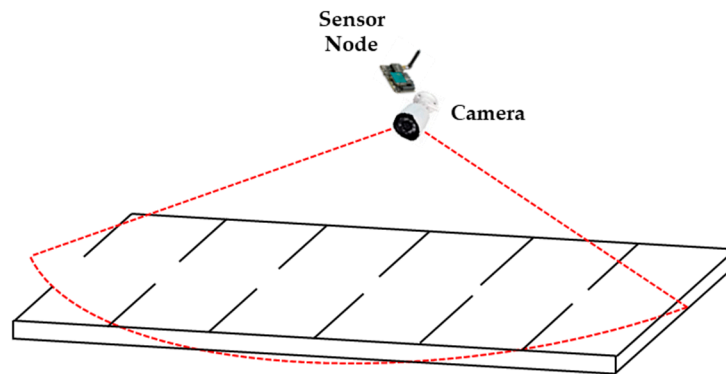


Figure 5. Evaluation scenario.

5. Numerical Results

In this section, we present the numerical results of the proposed algorithm. Every time a car arrives, the camera takes a picture and sends the picture to the Data Center that runs the proposed algorithm as described in the proposed algorithm in Section 3.2. To verify the performance of the proposed algorithm, the experiments were carried out at different times of the day in the morning, afternoon and at night. So it is possible to evaluate the proposed algorithm in the different environments of elimination. Figure 6 shows the processing time to recognize the plate numbers.

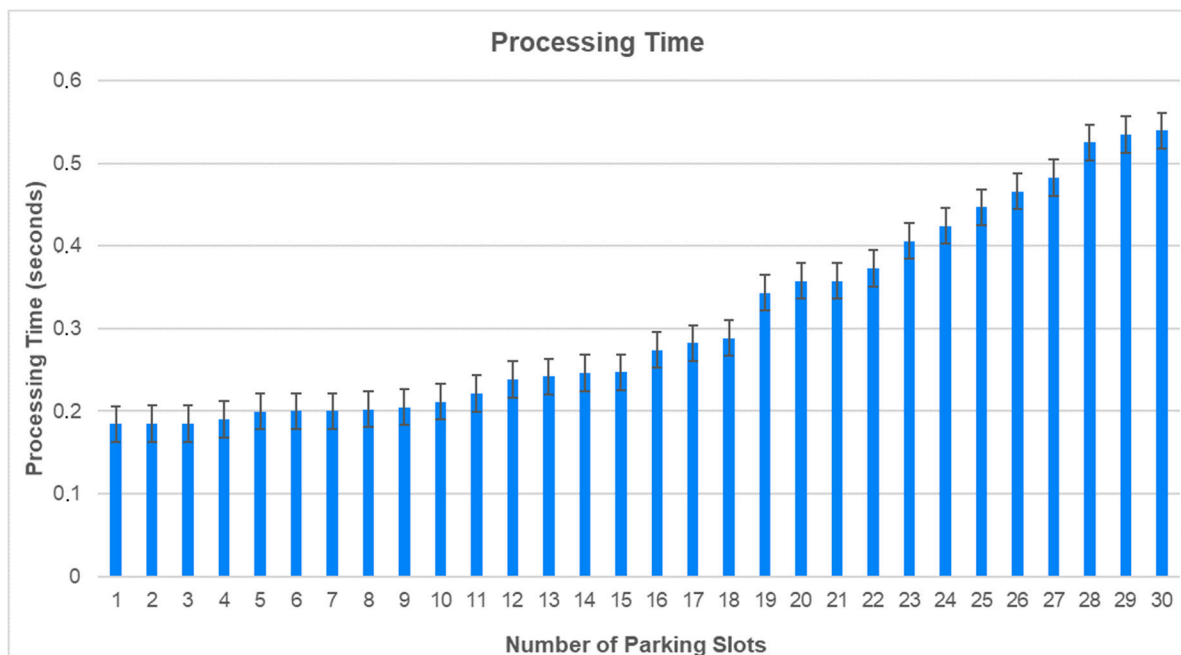


Figure 6. Processing Time.

The processing time starting from slot 1 to slot 18 was detected during the day (in the morning and on the afternoon), and the processing time from slot 19 to 30 was detected when the illumination started to get dark, and in this case, it used the parking illumination (in the evening). Figure 7 shows the average processing time.

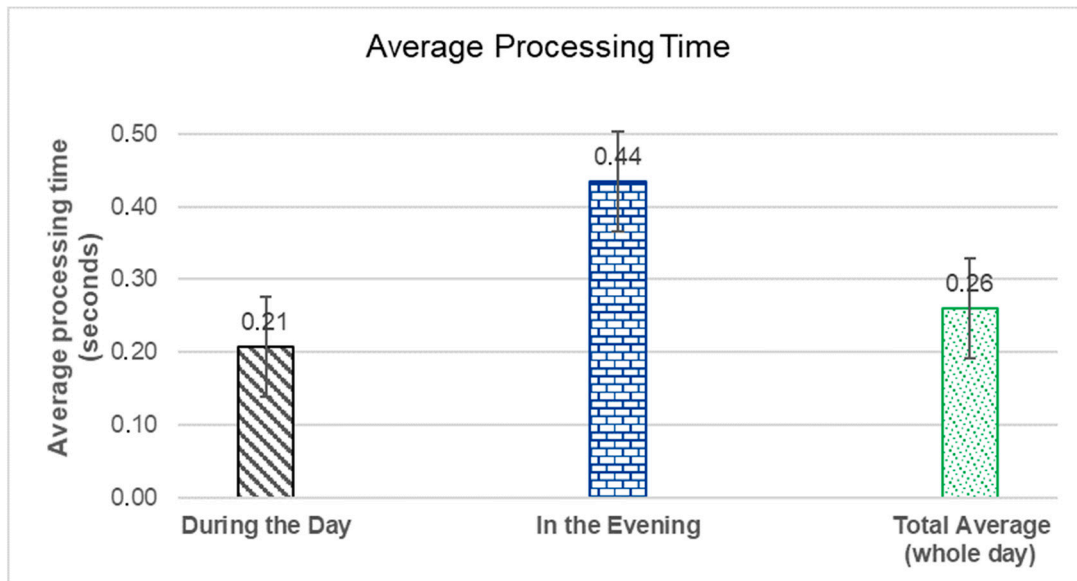


Figure 7. Average Processing Time.

As can be seen in Figure 7, during the day the processing average of the processing time is 0.21 s, and in the evening, the processing time is 0.44 s. The processing time increases in the evening because the illumination in the evening changes, becoming darker. However, the processing time, even in the evening, is low. Considering the illumination in both cases, it is possible to see that the average of the processing time during the whole day is 0.26 s, which is acceptable.

Figure 8 shows the accuracy of the proposed algorithm. Due to the variation of the environment illumination, we carried out the proposed algorithm changing the used filters. So, we also used the Gaussian Blur filter and the Filter 2D.

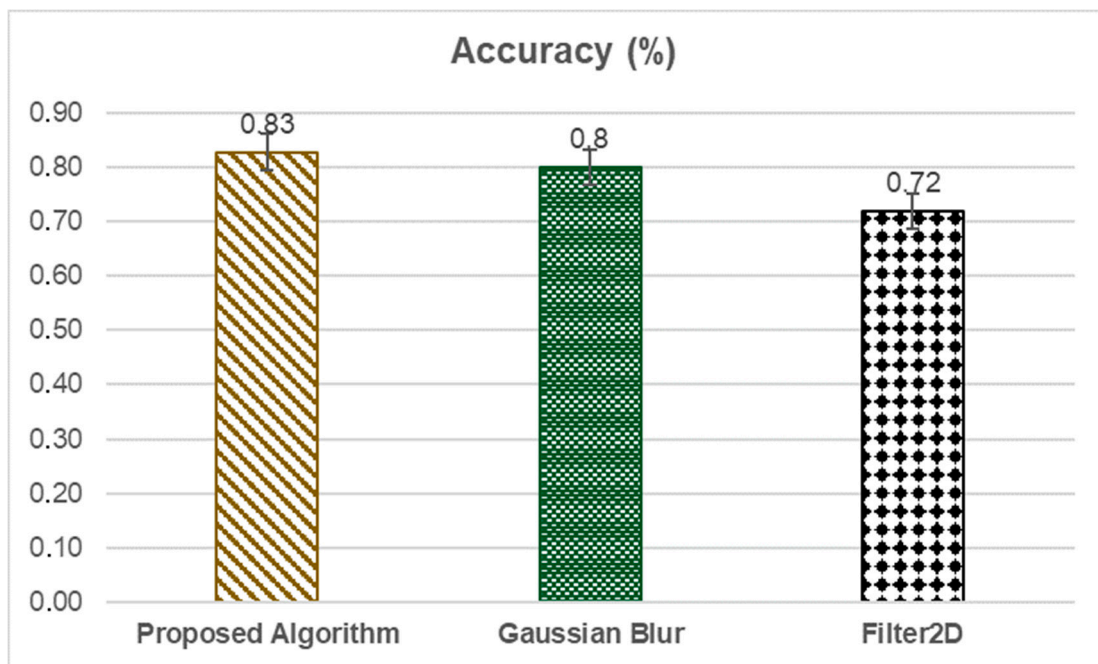


Figure 8. Accuracy of the proposed algorithm and the used filters.

As seen in Figure 8, the accuracy of the proposed algorithm is 83%, and the accuracy of the filters Gaussian Blur and Filter 2D is 80% and 72% respectively. So, the proposed algorithm shows a good result considering the filters Gaussian Blur and Filter 2D.

6. Conclusions

In this paper, a system that identifies in real-time vehicle license plate characters using optical character recognition (OCR) was proposed. The proposed system makes use of the intelligent parking service (SPANS) which is a framework used to detect available parking slots using computer vision functions. The proposed system uses the camera of the SPANS to get the images and information of the parking slots. Once a vehicle is detected, the proposed system takes a picture of the vehicle and uses this image to identify the license plate number of the vehicle. So, the identified number is stored on the system, and this information can be made available to public agencies such as traffic departments.

The algorithm used by the proposed system to detect the plate and to recognize the plate characters was described. The description starts with the plate image detection until the character recognition. Experiments were carried out at different times of the day. So, it was possible to evaluate the proposed system to take into account the variations in the environment illumination. So, different image filters were used in the proposed system, and the most robust filter was chosen to integrate the system.

Due to the variation of the environment illumination, the proposed system was evaluated using different filters on the pre-processing phase. The filters tested were the Gaussian Blur filter and the Filter 2D. The results showed that the proposed system had better performance when the bidirectional filter was used, detecting the plate characters on the environment with illumination variations. The processing time to make the plate recognition is low, and the proposed system showed a good performance in a real environment. However, it was possible to verify that the character recognition is sensitive to the environment illumination, and to improve the accuracy, new experiments will be done considering other types of filters as well as new deep learning methods will be implemented in the future works.

Author Contributions: Conceptualization, M.A.T.; Funding acquisition, R.I.M.; Investigation, M.A.T.; Methodology, N.d.V.D.; Supervision, M.A.T.; Writing—original draft, N.d.V.D.; Writing—review & editing, M.A.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Sao Paulo Research Foundation (FAPESP), grant number # 2019/26995-5, #2017/01055-4, and #2015/11536-4 and CNPq.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Detlev Mohr. *The Road to 2020 and Beyond: What's Driving the Global Automotive Industry?* Available online: https://www.mckinsey.com/~{}~/media/mckinsey/dotcom/client_service/Automotive%20and%20Assembly/PDFs/McK_The_road_to_2020_and_beyond.ashx (accessed on 9 September 2019).
2. Meneguette, R.I.; Geraldo Filho, P.R.; Guidoni, D.L.; Pessin, G.; Villas, L.A.; Ueyama, J. Increasing intelligence in inter-vehicle communications to reduce traffic congestions: Experiments in urban and highway environments. *PLoS ONE* **2016**, *11*, e0159110. [[CrossRef](#)] [[PubMed](#)]
3. Suryady, Z.; Sinniah, G.R.; Haseeb, S.; Siddique, M.T.; Ezani, M.F. Rapid development of smart parking system with cloud-based platforms. In Proceedings of the 5th International Conference on Information and Communication Technology for The Muslim World, Kuching, Malaysia, 17–18 November 2014; pp. 1–6.
4. Barone, R.E.; Giuffrè, T.; Siniscalchi, S.M.; Morgano, M.A.; Tesoriere, G. Architecture for parking management in smart cities. *IET Intell. Transp. Syst.* **2014**, *8*, 445–452. [[CrossRef](#)]
5. Mahalakshmi, S.; Tejaswini, S. Study of Character Recognition Methods in Automatic License Plate Recognition (ALPR) System. *Int. Res. J. Eng. Technol. IRJET* **2017**, *4*, 1420–1426.
6. Patel, C.I.; Shah, D.; Patel, A. Automatic Number Plate Recognition System (ANPR): A Survey. *Int. J. Comput. Appl.* **2013**, *69*, 21–33. [[CrossRef](#)]

7. Cheng, G.; Zhou, P.; Han, J. Learning Rotation-Invariant Convolutional Neural Networks for Object Detection in VHR Optical Remote Sensing Images. *IEEE Trans. Geosci. Remote Sens.* **2016**, *54*, 7405–7415. [CrossRef]
8. Cheng, G.; Zhou, P.; Han, J.; Xu, D. Learning Rotation-Invariant and Fisher Discriminative Convolutional Neural Networks for Object Detection. *IEEE Trans. Image Process.* **2019**, *28*, 265–278. [CrossRef] [PubMed]
9. Saha, S.; Basu, S.; Nasipuri, M. Automatic Localization and Recognition of License Plate Characters for Indian Vehicles. *Int. J. Comput. Sci. Emerg. Technol. IJCSET* **2011**, *2*, 520–533.
10. Khaparde, D.; Detroja, H.; Shah, J.; Dikey, R.; Thakare, B. Survey on Automatic Number Plate Recognition System. *Int. J. Comput. Appl.* **2018**, *69*, 28–32. [CrossRef]
11. Jing, Y.; Youssefi, B.; Mirhassani, M.; Muscedere, B. An efficient FPGA implementation of optical character recognition for license plate recognition. In Proceedings of the 2017 IEEE 30th Canadian Conference on Electrical and Computer Engineering, Windsor, ON, Canada, 30 April–3 May 2017.
12. Yimyam, M.; Ketcham, M. The Automated Parking Fee Calculation Using License Plate Recognition System. In Proceedings of the 2017 International Conference on Digital Arts, Media and Technology, Chiang Mai, Thailand, 1–4 March 2017.
13. Pan, M.; Yan, J.; Xiao, Z. Vehicle license plate character segmentation. *International J. Autom. Comput.* **2008**, *5*, 425–432. [CrossRef]
14. Neto, F.; Han, D.; Jeong, M. Online license plate matching procedures using license-plate recognition machine and new weighted edit distance. *Transp. Res. Part C Emerg. Technol.* **2012**, *21*, 306–320. [CrossRef]
15. Castro, M.; Teixeira, M.A.; Meneguetti, R.I. Automatic Parking Management System based on Wireless Sensor Network for ITS. In Proceedings of the 1st Workshop of Urban Computing—CoUrb, Campos de Jordão, Brazil, 15–10 May 2017.
16. Kaehler, A.; Bradski, G. *Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library*; O'Reilly Media, Inc.: Sebastopol, CA, USA, 2016.
17. Pi, R. Raspberry pi. 2017. Available online: <https://www.raspberrypi.org/> (accessed on 15 August 2019).
18. Namysl, M.; Konya, I. Efficient, Lexicon-Free OCR using Deep Learning. In Proceedings of the 15th International Conference on Document Analysis and Recognition (ICDAR 2019), Sydney, Australia, 20–25 September 2019.
19. Du, S.; Ibrahim, M.; Shehata, M.; Badawy, W. Automatic License Plate Recognition (ALPR): A State-of-the-Art Review. *IEEE Trans. Circuits Syst. Video Technol.* **2013**, *23*, 311–325. [CrossRef]
20. Tesseract. Available online: <https://opensource.google/projects/tesseract> (accessed on 10 July 2019).
21. Peixoto, S.P.; Cámara-Chávez, G.; Menotti, D.; Gonçalves, G.; Schwartz, W.R. Brazilian License Plate Character Recognition using Deep Learning. In Proceedings of the XI Workshop de Visão Computacional—WVC 2015, São Carlos, Brazil, 5–7 October 2015.
22. Silva, F.A.; Artero, A.O.; de Paiva, M.S.V.; Barbosa, R.L. Um Algoritmo Rápido para o Reconhecimento de Caracteres. In Proceedings of the VII Workshop de Visão Computacional, Curitiba Brazil, 22–25 May 2011; pp. 149–154. (In Portuguese).
23. Galindo, J.C.F.; Castro, C.L.; Braga, A.P. Sistema Automático para Reconhecimento de Placas de Automóveis Baseado em Processamento Digital de Imagens e Redes Perceptron de Múltiplas Camadas. In Proceedings of the 2016 XIX Encontro Nacional de Modelagem Computacional, João Pessoa, Brazil, 18–21 October 2016; pp. 19–21. (In Portuguese).
24. Naito, T.; Tsukada, T.; Kozuka, K.; Yamamoto, S. Robust license-plate recognition method for passing vehicles under outside environment. *IEEE Trans. Veh. Technol.* **2000**, *49*, 2309–2319. [CrossRef]
25. Shapiro, V.; Gluhchev, G.; Dimov, D. Towards a Multinational Car License Plate Recognition system. *Mach. Vis. Applications* **2006**, *17*, 173–183. [CrossRef]
26. Tomasi, C.; Manduchi, R. Bilateral filtering for gray and color images. In Proceedings of the Sixth International Conference on Computer Vision, Bombay, India, 4–7 January 1998.
27. Car Image, Placa Mercosul. Available online: <https://www.gazetadopovo.com.br/automoveis/governo-desiste-de-acabar-com-placa-mercosul-e-promete-melhorar-seguranca-6kye5lwot82kfv6qhsa7pez4v/> (accessed on 18 September 2019). (In Portuguese).
28. Teixeira, M.A.; Salman, T.; Zolanvari, M.; Jain, R.; Meskin, N.; Samaka, M. SCADA System Testbed for Cybersecurity Research Using Machine Learning Approach. *Future Internet* **2018**, *10*, 76. [CrossRef]

29. Meneguette, R.I.; De Grande, R.E.; Loureiro, A.A.F. *Intelligent Transport System in Smart Cities*; Springer: Cham, Switzerland, 2018.
30. Manliguez, C. Generalized Confusion Matrix for Multiple Classes. Available online: https://www.researchgate.net/publication/310799885_Generalized_Confusion_Matrix_for_Multiple_Classes (accessed on 25 August 2019).



© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).