Research Article
Open Access
CrossMark

A prototype of a car parking management service based on wireless sensor networks for ITS

Abstract

In this paper, we develop an intelligent parking service based on the wireless sensors for public or private spaces. The low-cost sensor node that will monitor the car park field. The nodes will capture images of the park field and send periodically to the intelligent transport system. The system can be accessed by the mobile devices to perform various management functions, such as finding vacant parking lots, and statistic report. We have implemented a prototype of the service using Open CV library. The service evaluation demonstrates the effectiveness of our design and implementation of our solution, showing that the proposal service can manage parking places without impairs other services that can be performed in the intelligent transport system.

Keywords: parking service, intelligent transport system, cv library, monitor

Volume 2 Issue 3 - 2017

Marcelo RO Castro, Marcio A Teixeira, Luis HV Nakamura, Rodolfo I Meneguette
Federal Institute of Sao Paulo (IFSP), Brazil

Correspondence: Marcelo RO Castro, Federal Institute of Sao Paulo (IFSP), Brazil, Email Marceloddcastro@gmail.com

Received: April 18, 2017 | **Published:** May 09, 2017

Introduction

The wireless sensor networks (WSN) has been received considerable attention from the research, governments and companies due to their potential of providing an easy and cost effective solutions in many different areas.[1] Therefore, WSN can be integrated with other technologies to assist services from intelligent transport system (ITS).[2] ITS is the junction of various technologies focus on providing services and applications that will monitor, manage the transportations system, and also make the drive more comfortable and safety.[3,4] According to the McKinsey prediction,[5] the number of the vehicle will increase from 1.01 billion in 2013 to 1.32 billion by 2020. This increasing will impact on the traffic congestion due to the demand of road space at peak times,[6] as well as, it will generate acute parking shortages in cities as demand again exceed supply. Currently, the method of finding the parking space is by brute force method, where the driver is seeking the vacancy through the streets near the destination or even remote. This method considers the knowledge of the place (experiences) and with the luck of the drivers. Furthermore, it can increase the traffic congestion due to the velocity that drivers used when they are looking for a parking space.[7] An alternative is to find a predefined car park with high capacity. However, this is not an optimal solution because the car park could usually be far away from the user destination.[8] Thus, computational systems could manage the parking lot through a combination of different technology, i. e. The ITS could prove a service that detects, manage and inform the users about the parking lot available in order to release the time of find a parking and the congestion. In the literature, there is some works that address the free parking lot problem.[1,2,8–13] However, some of these works use external hardware to support their application. Furthermore, these works include a RFID or other device to identify the packet lot and the vehicle. Thus, the vehicle needs more embedded devices to support the management of parking lot. In this paper, we proposed a new service that does not necessary other devices in the vehicle to manage the parking lot available in the cities. To detect available parking lot, the service use images that can be obtained by the city camera system or by the proposal wireless node device that will make snapshots of the parking field and will send to the data center. The data center will process the image an it will indicate available parking lot for the user through the mobile application. The main contributions of this paper are:

a. Developing and analyzing of image processing service to detect available parking lot spaces in an indoor and outdoor places
b. Developing an Intelligent transport system data center to manage and control this service
c. Developing a mobile device to notify the users about the available parking lot
d. Developing cheap device that will captured the images about the parking fields

The rest of this paper is structured as follows. In the next section, there is an overview of the main existing approaches for vehicle congestion detection in VANETs. The proposed solution is described in Section 3, while a detailed performance evaluation and results are shown in Section 4. Finally, Section5 summarizes the conclusion and make suggestions for future work.

Related works

In the literature there are some works have addressed the Car Parking Management system for public or private spaces. Li et al.,[9] proposed a method for car searching system based on a smartphone. This method uses some function of indoor localization and path navigation to detect and search vehicle into a large park lot. The author developed an application based on the QR code and mobile application. The QR code is used to format the information that assists the search and detect the parking spot available, these data consist of the parking lot, and floor and parking location are encoded into QR code. The mobile applications are: offline map, scanning the nearly QR code to record the parking location, planning the optimal car-searching path, real-time navigation. Thus, the drivers can scan and decode QR codes using a smartphone, and the localization application to find the parking lot and their vehicle in a large park. Pham et al.,[8] Proposed a parking system that helps drivers find a free parking space. This system is based on Internet of Thing (Io T) and use a data center serves as a cloud server to calculate the costs of a parking request. The author also use a Local Unit that to res the information of each parking space and the located in each car park. This local unit include control unit that consists of an Arduini module, which is connected using an RFID reader to verify and authenticate driver information and calculate the free spaces in each car park and identify the vehicles.

Moreover, it also include a Screen that show information about the capacity of the local car park, the total f free spaces, the status of the RFID tag check, and a mini map of the local car park. Yeh et al.,[10] proposed a system of city parking integration. The system combine smart mobile devices, cloud computing technologies, and the global positioning system (GPS). Furthermore, the author developed an application that makes reservation of park lot as well as provider a parking navigation services. When the driver turn on the application, it send the user location information to the cloud computing system. The cloud system also receives the information the parking information that that make through the scan of RFID system in the parking lot. After that the system analysis, send the five closest parking lots with have available parking spaces to the driver application, allowing them to choose the most suitable parking space. Barone et al.,[2,11] proposed a parking model referred to as intelligent parking assistant (IPA). This approach will allow that drivers make some reservation of the parking lot at destination before their departure. In addition, it will provide information about on-street parking lot availability. IPA is based on SPARK system that uses the RFID to indentify the vehicle. The RFID allows that IPA gets into the parking spots after this has been reserved without an internet connection. IPA allows that other drives make a reservation to other people, simply handing small radio frequency identification (RFID) tag. Sheelarani et al.,[12] proposed application that supported the drivers to park their vehicles by finding an empty parking lot. This application is based on Android platform and uses embedded hardware to assist the application. In this approach, the authors use an Infrared Sensors (IRS) to find availability of the slot for parking the vehicle. Furthermore, the vehicle consists of a RFID tag that is used to identify the user. Thus, the drivers can book a park lot in a field by their RFID tag identification. Our proposal is based on the image processing methods to detect available parking lot in the inside or outside environment. Unlike some other methods, in the proposal method does not necessary into hardware or other embedded device into the vehicle, it uses images captured by the wirelesses sensors node to indicate available lot to the driver. Furthermore, the service will be access anywhere and anytime, the system only needing the connection to the data center (cloud).

The car parking management service based on wireless sensor networks for it's

This section describes a solution for the detection and manage available parking lot in a city, called SPANS – Smart Parking Service. S PANS is a service based on image process that will detect available parking lot. Furthermore, these services integrated into an intelligent transport architecture that will provide the structure for detect, manager and notify the drives about this lot. These infrastructure aims to avoid heavy traffic in areas that have parking place thereby reducing the time and the consumption of the fuel and pollution caused by vehicles that search an available lot. The infrastructure uses sensors to perform the sensing of a parking place and cameras to record the activities. Sensors and cameras will be used to detect significant information from the parking so that it is allowed to infer if there are some lots available. Furthermore, this infrastructure consists of a data center (cloud) that provide a good mechanism for data abstraction and image processing, as well as a good communication mechanism between drivers and the sensors. Moreover, the data center will offer a security mechanism which difficult other people have access to the information. The drives will access the data center through a mobile application that can run on a smartphone or tablet. Figure 1 shows and abstraction of the proposal infrastructure used to detect and notify the user about the free parking lot. Therefore, the data center from time to time or when the sensor detects a change in the environment, receives information about the parking place. This information is processed by SPANS that will detect if any parking lot is available. After his processing, the system will provide this information about the available places that the driver can be used to the driver's application. Once the driver accesses your mobile application, the information will show the place that the drivers can stop their near of their destination. In the following subsections, we describe how the proposal infrastructure was developed as well as the services and equipment. Furthermore, we describe the sensor node in case that the place does not have a structure of cameras that will capture images of the parking lots and information about the available lot.

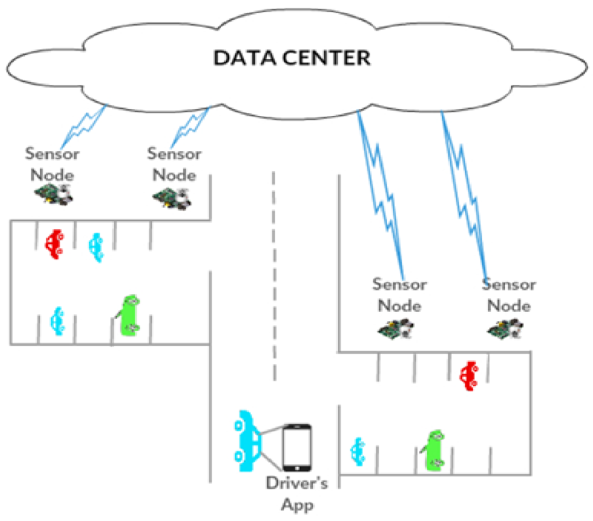


Figure 1 An Abstraction of proposal infrastructure.

Prototype of data

Center For the development of this work, the data center are implemented as REST Full web services that has the Internet connection to receive requests from the driver's applications as well as the receive the images collected by the sensors. The data center consists of an abstraction and data processing module execute the tasks in the central computer, by obtaining data from the sensors and then analyzing and saving the min the database. The data are analyzed by SPANS that will detect and indicate the places where are the available parking lot. After that, this information is available to the driver's application. A Dell processor to support the data center. In this processor, we installed a My SQL database, Apache CXF framework to create web services, as well as, we used Jetty as the web server and servile container together with our Java algorithms.

Prototype of wireless sensor node

Often places that offer parking do not have a surveillance system that monitors the entrance and exit of vehicles as well as parking. In other words, this establishment does not have a camera system capable of verifying the entry and exit of vehicles in a particular parking space. Thus, we have developed a solution to be able to monitor these environments. This device besides sending from time to time images of them monitored place it will also be covered to determine the exit and entrance of a vehicle and a parking space. In other words, the sensor will capture a change of scenery in the environment and will capture the new image, thus it will be possible to deter the movements of a particular parking space. The wireless sensor node is based on a Raspberry Pi model B. Raspberry Pi consist of an USB wireless network interface and a traditional Webcam. Furthermore the Raspberry used a Raspberry operational system, a Python program to read the data from the Raspberry GPIO (General Purpose Input/output) pins and Motion for monitoring video signals from the Webcam. We choose the Motion because this software can be used as a motion detection which writes a file every time that there is a change in the image frames. To achieve this, we need to set a time interval in seconds of no motion detection triggers the end of an event. We set the Motion to take a picture after 10 seconds that it detects a change in the environment. This time was chosen to give the vehicle time to get out or park the vehicle.

The parking lot information

The parking lot area is splitting in several parking lot places as shown in the Figure 2. Each parking lot space has the following information:

ID Park: Parking lot place identification

a. Desk: Parking lot space description
b. X: Coordinate ”X” of the parking lot space in the parking lot area
c. Y: Coordinate ”Y” of the parking lot space in the parking lot area
d. Width: Width of the parking lot space
e. Height: Height of the parking lot space
f. Status: Store the current status of the parking lot space. 0 is available; 1 is busy

The parking lot information is storage in a database. Once defined the parking lot places, using a camera, the system will get an image named ”Base Image” of each parking lot space taking into account the X and Y coordinates and the width and height values. The”Base Image” will be storage into the database and will be used to verify whether the parking lot place is available or busy. Initially, all parking lot places will be considerate available as showed in the Figure 2.

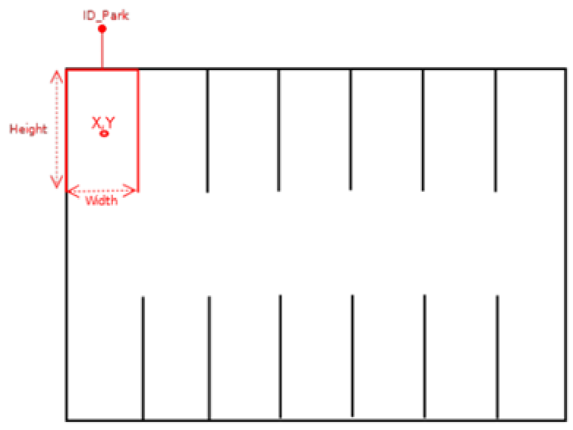


Figure 2 Parking lot information.

Image processing

Once the data center receives the pictures of the parking place, it starts SPANS. SPANS will perform the processing of the image with the purpose of verifying the available lot that the driver can park. For achieve this, SPANS needs to follow some steps to allow an efficient detection in indoor and outdoor environment. Figure 3 that describe the steps that SPANS process need to make in order to can have the perception of available parking lot independently of a park in a closed area or in the open air.[13]

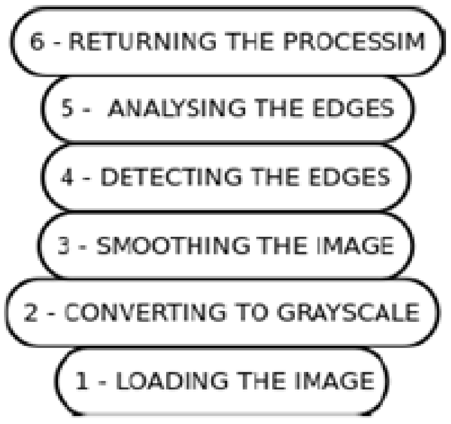


Figure 3 The process image steps that the SPANS needs to follow.

Thus, the first steps the SPANS will load the image that the data dented received from the sensors node. After that the SPANS will convert the image in a matrix MxN, where the row and the column indicate as spatial coordinates in the image, the pixel position. Figure 4 show a color image and its matrix pixel red, green and blue. Posteriorly, the system will transform the color image to the grayscale image. For achieving this, each pixel of the image is read and a grayscale color is calculated in that pixel. The Figure 5 describes a Java algorithm that receives a color image and returns a grayscale image. After this process, the image is passed through a filter called Glaussian Blur[14] to smooth the image. This filter is a low pass filter that will let pass the low frequencies, but it will eliminate the values related to the high frequencies. Thus, the effect of this filter is a smoothing of the image, since the high frequencies that correspond to the abrupt transitions are attenuated. Smoothing tends to reduce noise in images.[15] The Gaussian Blur uses a Gaussian function that can be described as:

$$G(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}} \qquad (1)$$

When we consider the function in two dimensions it results in the product of two equations of one dimension:

$$G(x) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \qquad (2)$$

Where x is the distance from the origin on the x-axis, y is the distance from the origin on the y-axis and sigma is the standard deviation of the Gaussian distribution. Choosing values for alpha is possible obtained arrays that will be used as the kernel to convolve the image. After the convolution, it is possible to verify a smoothing of the image, as we can see in Figure 6. Other technique used is the edge detection that highlights edge contours and also amplifies scene noise and, in large part, edge operators, enjoy some kind of image smoothing before the differential operation.[16] In this paper, we use the canny algorithm[17] which is a first derivative Gaussian operator, smoothing the noise and locating the edges. For achieve this, this algorithm uses three rules:

a. Detection: ability to locate and mark all the existing borders
b. Location: reduce the distance between the true border and the detected border
c. Answer: there is only one response for each edge

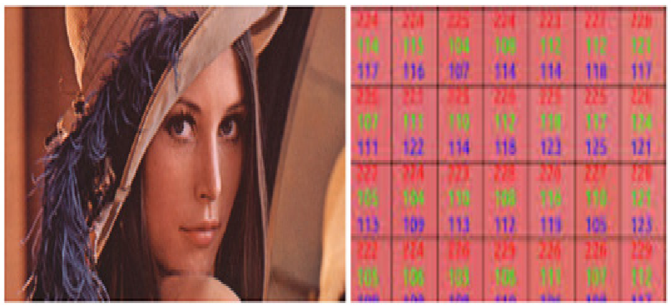
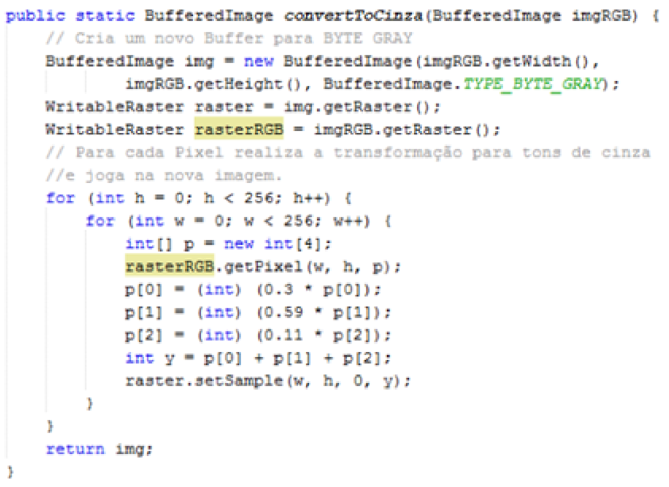


Figure 4 Representation of pixels in colored digital images.



```
public static BufferedImage convertToCinza(BufferedImage imgRGB) {
    // Cria um novo Buffer para BYTE GRAY
    BufferedImage img = new BufferedImage(imgRGB.getWidth(),
            imgRGB.getHeight(), BufferedImage.TYPE_BYTE_GRAY);
    WritableRaster raster = img.getRaster();
    WritableRaster rasterRGB = imgRGB.getRaster();
    // Para cada Pixel realiza a transformação para tons de cinza
    //e joga na nova imagem.
    for (int h = 0; h < 256; h++) {
        for (int w = 0; w < 256; w++) {
            int[] p = new int[4];
            rasterRGB.getPixel(w, h, p);
            p[0] = (int) (0.3 * p[0]);
            p[1] = (int) (0.59 * p[1]);
            p[2] = (int) (0.11 * p[2]);
            int y = p[0] + p[1] + p[2];
            raster.setSample(w, h, 0, y);
        }
    }
    return img;
}
```

Figure 5 Convert colored image to grayscale in Java.

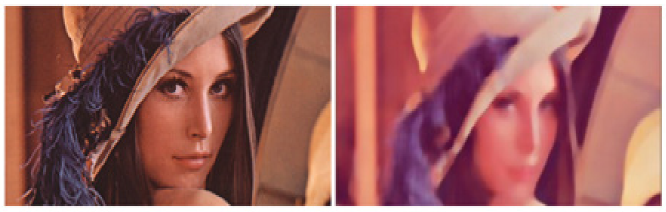


Figure 6 Smoothing of the image.

Figure 7 show the comparison of the original image with the image hat pass by the Canny. For implemented all this process we used the Open CV library[18] that provides several methods to be used in the image processing. The proposal algorithm firstly reads the images (lines 2-3 in Algorithm 1). The algorithm converts the read images to grayscale (lines 4-5) through the cut Color function. After that, it applies filters to remove the imperfections and noises (lines 6-7) by Blur function. So, the algorithm detects the number of borders in the images (lines 8-9) using Canny and comparing them (10-12) through the find Contours that retrieve contours from the image. The contours are used to detect the objects in the images. If the number of the Current Image is greater than the Base Image, the system set the status of the parking lot place as busy. Otherwise, the system set the status of the parking lot place as available (lines 13-16). This information is updated into the database (line 17) and is used by the mobile application. The results of this algorithm we can see in the Figure 8 that shows the images gotten from the implementation of the proposed algorithm using the Open CV functions.

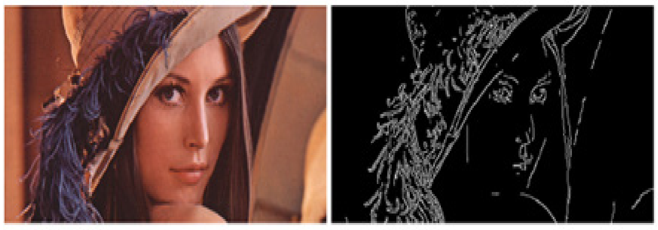


Figure 7 Result of applying Canny to an image.

Algorithm 1 Image Processing Algorithm

```
Read Base_Image
Read Current_Image
Transform the Base_Image to Grayscale
Transform the Current_Image to Grayscale
Apply Filter to Remove Noise (Base_Image)
Apply Filter to Remove Noise (Current_Image)
Edge Detection(Base_Image)
Edge Detection(Current_Image)
Number_Boder_base = Extract Border (Base_Image)
Number_Boder_Current = Extract Border (Current_Image)
Compare Borders (Number_Borders_base, Number_Boder_Current)
if (Number_Boder_Current > Number_Boder_Current ) then
    Status[ID_Park]= Busy
else
    Status[ID_Park]= Availabel
end if
Update Database
```

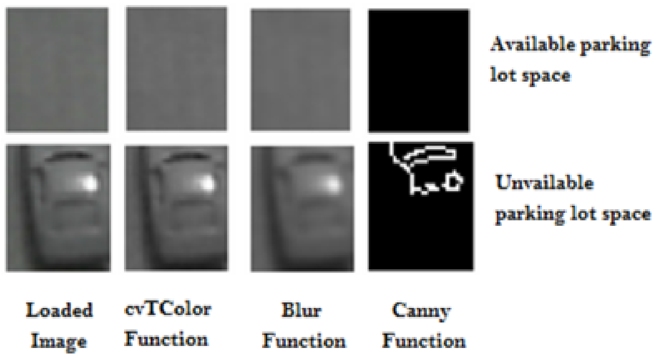


Figure 8 Image processing algorithm.

Driver's application

In order for the drivers to have access to information about parking lots available near the place of destination, we have developed an application for the android platform that will perform a secure communication with the data center through SSL protocol. This application also has a screen to show the map of parking spaces through the processing of the photos taken by the sensors so that the user can check the best available space for it.

User case

In this section, we describe more detail about the proposal service. SPANS were implemented in C++ language to keep a compatibility of among the elements that involves the service. However, the implementation of the wireless sensor node also uses python to facilitate the manager of the embedded device. Moreover, the system uses the Open CV library not only to implement the image processing but also to create the mechanism to control the webcams remotely. The Figure 9 shows the class diagram of SPANS service in the data center. The Camera class was created to encapsulate the Open CV class called Video Capture which provides an API for capturing video from the webcam embedded in the sensors node, as well as, allowing SPANS reading video files and image sequences in the database or any other monitoring system. Thus, the class camera is derived from the class Video Capture that defines methods to access sequential of the images from the database. The Cap Thread class is one of the most important of the application because it is it that performs all the processing weights of comparison between the images to find out if the monitored lot is idle or not. The rendering compares all the vacancy images that were saved to the local repository with the current image extracted from the frame that is constantly updated by the camera. When instantiating a new Cap Thread it passes as an instance of a Camera class in its constructor from there the Cap Thread will have access to all the vacancies that were loaded in the Camera class. For the infrastructure has a good efficiency, it initially takes a picture of the parking lot. This photograph is requested only once in order to register the coordinates of each parking space, in addition this image will be used as the default image (Base Image) of that establishment. So, the infrastructure can subsequently perform the detection of available parking spaces. The Figure 10 shows this first stage of the system, in which there is a vacancy registered in the system, represented by the yellow square. This registration of vacancies can be carried out by the owner of the establishment. Thus the system will provide support not only to the users and managers of the city, but also to the owners of these establishments.

Once defined the parking spaces, the system starts the monitoring of the parking spaces through the new images coming from the sensor nodes. The data center will use the Base Image and the image obtained real-time from the sensor to detect the available parking lot. The service draws a green rectangle in all available parking spaces. Whether the system detects that a parking space has been occupied, it updates the status information of the parking space into the database and draws a red rectangle in the parking place busy as we can see in the Figure 11. The information stored into the database about the parking space is accessed by the mobile application that will show the information to the users using a friendly interface. The Figure 12 illustrates the interface showed to the users. The mobile application has a grid interface where will showed the current status of each parking lot place defined in the system.

Results

In order to evaluate the SPANS service, we used the developed infrastructure to make tests were made using a scale model of the parking area where were defined 20 parking lot places. Each parking lot place was inserted into the database with the available initial status. For this, we created a model that represents the parking place of Federal Institution of So Paulo, Catanduva, Brazil. The Figure 13 shows the result of the image processing take into account the scale model used in the tests. This test will evaluated the time that the service will take to processes the images based on the number of lots that the parking has. Therefore, we want to verify the time spent to find the parking lot places available. The Figure 14 shows the processing time spent to find the available places. As can be seen in the Figure 14, the average time spit to find an available place in the parking lot area is 0.036 seconds. This means that the time spent to find the available places was proportional for all parking area. The system could find the parking lot places available with 93% of accuracy in the tested environment. Thus, SPANS can manage parking places without impair other services that can be perform in the intelligent transport system.

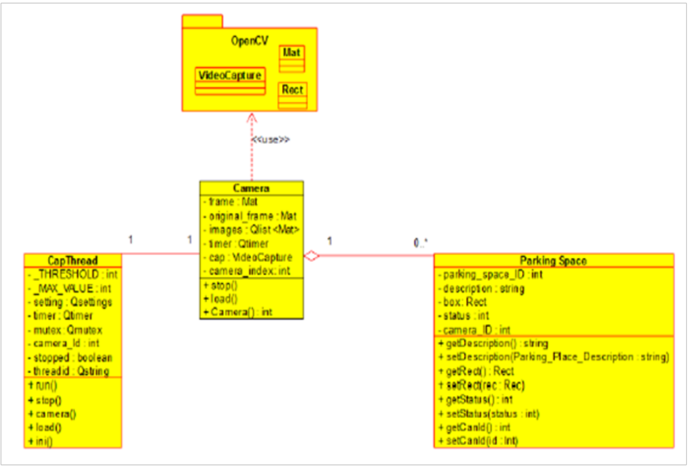


Figure 9 Diagram of classes using UML of SPANS.

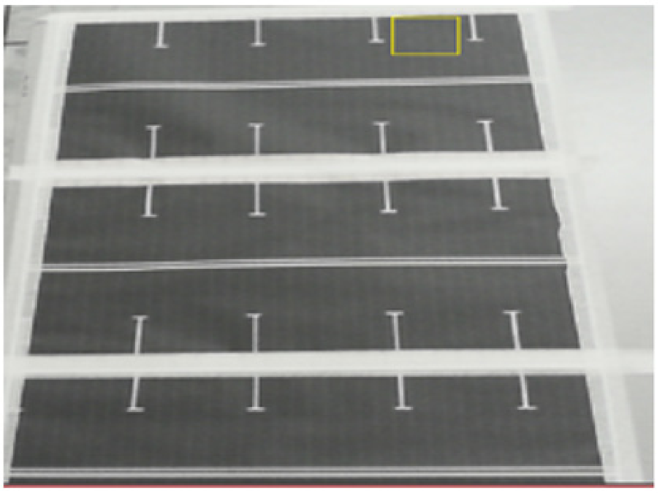


Figure 10 Scale model of the parking area.

Conclusion

In this paper, we have proposed a service that will detect and manage available parking lot in a city. The proposed service is based image process that will detect available parking lot. Furthermore, this service is provided by an intelligent transport architecture that aims to avoid heavy traffic in areas that have a parking place. Thus, reducing the time and the consumption of the fuel and pollution caused by drivers that are searching for an available lot. Moreover, we made a user case that shows how to developing the proposal solution, as well as, show the performance of this service. The results showed that SPANS can manage parking places without impair other services that can be performed in the intelligent transport system. The SPANS also showed a good percentage of accuracy in the tested environment about 93%. As future work, we intend to create a sub module so that the owners of this parking can serve with skink to the sensor nodes, as well as, facilitate the management of your establishment. This sub module will also allow a decrease of the banking width used in the communication between the skin and the data center since the sub module only needs to pass a text file from time to time indicating the available lots and not the image. Also intending to install this structure not only in the university but in the parking lots near the university. In addition, perform tests that evaluate the communication capabilities of the network.

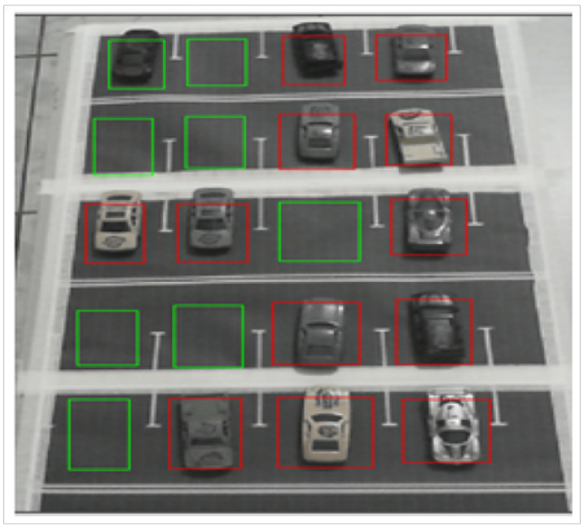


Figure 11 Example of the system running.

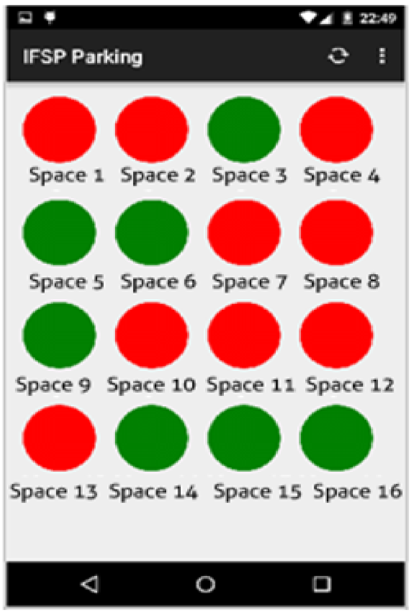


Figure 12 Interface of the mobile application.

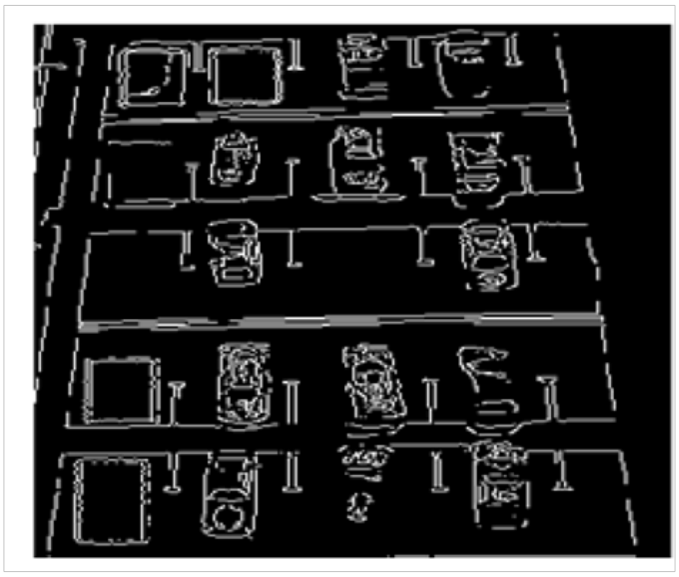


Figure 13 Scale model of the parking area after the image processing.

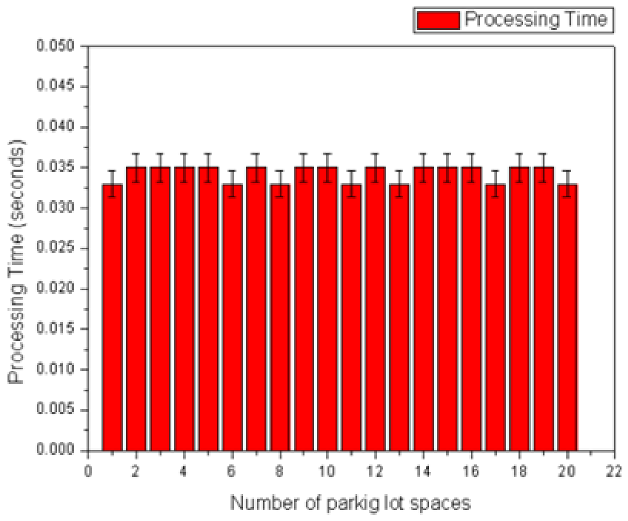


Figure 14 Scale model of the parking area after the image processing.

Acknowledgments

None.

Conflict of interest

Author declares that there is none of the conflicts.

References

1. Suryady Z, Sinniah GR, Haseeb S, et al. Rapid development of smart parking system with cloud–based platforms. *Information and Communication Technology for the Muslim World (ICT4M)*. 2014. p. 1–6.

2. Barone RE, Giuffre T, Siniscalchi SM, et al. Architecture for parking management in smart cities. *IET Intelligent Transport Systems*. 2014;8(5):445–452.

3. Meneguette RI, Filho GP, Guidoni DL, et al. Increasing intelligence in inter–vehicle communications to reduce traffic congestions: Experiments in urban and highway environments. *PLoS One*. 2016;11(8):e0159110.

4. Nellore K, Hancke GP. A survey on urban traffic management system using wireless sensor networks. *Sensors*. 2016;16(2):157.
5. Mohr D, Muller N, Krieg A, et al. The road to 2020 and beyond: what's driving the global automotive industry? *Japan Automobile Manufacturers Association*. 2013;28(3):31.
6. Meneguette RI. A vehicular cloud–based framework for the intelligent transport management of big cities. *International Journal of Distributed Sensor Networks*. 2016;12(5):8198597.
7. Suhr JK, Jung HG. Automatic parking space detection and tracking for underground and indoor environments. *IEEE Transactions on Industrial Electronics*. 2016;63(9):5687–5698.
8. Pham TN, Sai MF, Nguyen DB, et al. A cloud–based smart–parking system based on internet–of–things technologies. *IEEE Access*. 2015;3:1581–1591.
9. Li J, An Y, Fei R, et al. Smartphone based car–searching system for large parking lot. *Industrial Electronics and Applications (ICIEA)*. 2016. p. 1994–1998.
10. Yeh Her–Tyan, Chen Bing–Chang, Wang Bo Xun. A city parking integration system combined with cloud computing technologies and smart mobile devices. *Eurasia Journal of Mathematics, Science & Technology Education*. 2016;12(5):1231–1242.
11. S Huang, H Fu. Design of embedded parking management system. *Communication Problem–Solving (ICCP)*. 2016. p. 1–2.
12. Sheelarani P, Anand SP, Shamili S, et al. Effective car parking reservation system based on internet of things technologies. *Futuristic Trends in Research and Innovation for Social Welfare (Startup Conclave)*. 2016. p. 1–4.
13. Ersoy FT, Hasker K, Inci E. Parking as a loss leader at shopping malls. *Transportation Research Methodological*. 2016;91(98):112.
14. Hummel RA, Kimia B, Zucker SW. Deblurring Gaussian blur. *Computer Vision, Graphics, and Image Processing*. 1987;38(1):66–80.
15. Arulampalam MS, Maskell S, Gordon N, et al. A tutorial on particle filters for online nonlinear/non–gaussian bayesian tracking. *IEEE Transactions on Signal Processing*. 2002;50(2):174–188.
16. Canny J. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 1986;8(6):679–698.
17. P Bao, L Zhang, X Wu. Canny edge detection enhancement by scale multiplication. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2005;27(9):1485–1490.
18. A Kaehler, G Bradski. *Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library*. 2016.