Smart Parking System Through Image Processing

Ali Taibi[#1], Farouk Jeffar[#2], Yasser Bennani[#3]

[#]*School of Science and Engineering, Al Akhawayn University in Ifrane, Morocco*

[1]A.Taibi@aui.ma

[2]F.Jeffar@aui.ma

[3]Y.Bennani@aui.ma

***Abstract*— This work is an attempt at developing a robust, sustainable and fairly easy to implement solution for the increasingly problematic matter of parking at Al Akhawayn University in Ifrane. This work tackles mainly the pertinence of using image processing and computer vision in vehicle detection and ultimately in achieving the objective of recognizing free parking spots using only surveillance cameras as data acquisition hardware.**

***Keywords*— Smart Parking, Image Processing, OpenCV, Python, Binarization, Canny edge detection, SOBEL, Laplacian, Grayscale, Region of Interest, Histogram, Histogram Equalization, Adaptive Histogram Equalization, Contrast Limited Adaptive Histogram Equalization, HSV, Image Optimization, VueJS, Firebase, OpenWeather API.**

Introduction

A smart parking system facilitates the management of a parking facility and makes the process easier and more convenient for drivers and managers of the parking. Managing the parking can be done through hardware only or by adding software. Our idea is to use Digital Image Processing to remove the hassle of using sensors and thus decreasing cost, our goal is to shift to a more software oriented parking management. Eliminating sensors from the parking process would result in reduced mechanical and electronic liability as only a surveillance cameras would be used. Our designed DIP algorithms would differentiate between full or empty spaces and then send the information to drivers and managers.

Problematic

Al Akhawayn car park is not efficiently managed. Students, Staff, and Professors lose considerable amounts of time looking for parking spots during busy times of the day. Consequently, drivers keep going around looking for a spot. This behavior has a bad repercussion on safety, time efficiency, and ecology.

- Safety: drivers are not fully focused on the road but rather focused on finding free spots which may cause dangerous hazards.
- Time efficiency: Students and faculty might lose time looking for parking spots and thus be late for their duties.
- Ecological: Instead of going directly to the free parking lot, drivers might have to drive unnecessary distance to find a spot which increases fuel consumption (Many gear shifts and accelerations) and thus more CO2 emissions.

Suggested Solution : Why Image Processing?

When it comes to parking space management, the more conventional way of implementing a smart system is the combination of sensors and small computing units in order to form a functional solution. In theory, this kind of systems is simple to implement however, the solution discussed in this work has the following advantages :

- Infrastructure: the already existing infrastructure of surveillance cameras in every parking space is an important asset.
- Cost: using only cameras remains a less expensive prospect in comparison with a set of sensors along with the necessary computing units.
- Practicality: there will be no need to install any further hardware or make any modifications to existing hardware.
- Performance: most conventional sensors suffer in unfavorable weather conditions such as rain. Whereas cameras are not affected.
- Maintenance: physical sensors are more prone to failure and have to be replaced frequently in order to maintain their functioning.

- Expandability: unlike the solution discussed in this work, sensor based systems are a challenge to expand and scale up from one parking space to a bigger set of parking spaces in terms of hardware and software as well.

Methodology of Use

The suggested solution is based on the following process:

1. Image/ Video Acquisition.
2. Image Enhancement and Optimization.
3. Application of object detection algorithm.
4. Retrieval and processing of algorithm results.
5. Display of relevant information to the end user.

Process of Optimizing the Image

Before starting with the detection algorithm, the captured image from the camera goes through many essential optimizing stages that are essential for applying the edge detection techniques.

Fig. 1 parking original image

A. Gaussian Blur

Gaussian Blur (or also known as Gaussian Smoothing) is a technique in image processing that is used to generate a blurred version of the original image using a two-dimensional Gaussian Function of the form:

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

Where σ is the standard deviation and the mean is assumed to be equal to 0. By convolving each pixel the original image with a Gaussian kernel of size 5x5 of the form:

$\frac{1}{273}$

1	4	7	4	1
4	16	26	16	4
7	26	41	26	7
4	16	26	16	4
1	4	7	4	1

Fig. 2 5x5 Gaussian Kernel [3]

we get the following image:

Fig. 3 Parking image with gaussian blur applied

Applying a Gaussian filter on the original image reduces substantially the amount of noise present in the video capture. It also helps in reducing unnecessary details which might hinder the main algorithm from detecting cars [1].

B. Grayscale

A grayscale image is an image where the color value of each pixel is a shade of gray that corresponds to the amount of light on that particular pixel. The contrast offers 256 different shades of gray from black to white. Since the image that is read from the video source use BGR color format, the conversion to grayscale is done using this formula:

$$Y = 0.114\,B + 0.587\,G + 0.299\,R$$

where Y is the luma of the image.

Converting the blurred image to grayscale yields the following image:

Fig. 4 Parking image with grayscale applied on gaussian

The conversion to grayscale is an important step in order for edge detection to work, and using grayscale image processing speed is faster and color information which is useless is discarded [2].

C. Histograms and Histogram Equalization

A histogram is a plot of the gray levels vs. the occurrences.

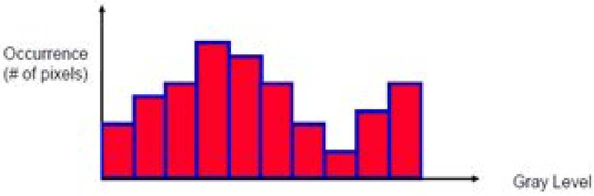

Fig. 5 Histogram Example [10]

Normalized histogram can be thought of as an estimate of the probability distribution of the continuous signal amplitude. Brighter images have a histogram concentrated at the higher intensity values and high contrast images have a flat histogram. So to overcome the shadow problem, flattening and thus brightening the frames through histogram equalization is essential. For a better visual discrimination of an image we would like to re-assign gray-levels so that gray-level resource will be optimally assigned.

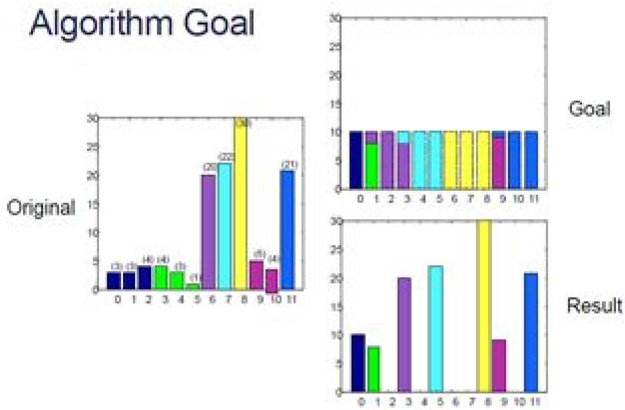

Fig. 6 Histogram Equalization desired output [10]

Adaptive histogram equalization is an image processing technique used to improve contrast in images. It computes several histograms; each corresponding to a distinct section of the image and uses them to redistribute the lightness values of the image. Moreover, it helps enhancing the definitions of edges and over Amplifies noise. CLAHE, also known as Contrast Limited Adaptive Histogram Equalization, is a variant of Adaptive Histogram Equalization. “In CLAHE, contrast limiting is applied. If any histogram bin is above the specified contrast limit, those pixels are clipped and distributed uniformly to other bins before applying histogram equalization. After equalization, to remove artifacts in tile borders, bilinear interpolation is applied.”(From OpenCV documentation.)

Fig. 7 Image after applying CLAHE

D. Setting ROI

A region of interest (ROI) is a portion of an image to which we apply filters and various image processing techniques in order to perform operations on. In the context of this work, each parking spot in the footage is a region of interest. The algorithm for detecting if a parking spot is occupied or free should be applied to all the regions of interest. A region of interest is generated by creating a binary mask image in which pixels that belong to the ROI are set to 1 and the pixels outside are set to 0. Each region of interest will be drawn using 4 points that will generate a bounding polygon that will be stored in a file. The Regions of interest's polygons indices will be comprised in a JSON file that will be loaded at the start of the detection program's execution.

Detection Algorithms: Techniques Explored

To fulfill the goal of object identification and more specifically the detection of vehicles, many techniques had to be explored in order to recognize the advantages and drawbacks of each one and determine the most adequate approach for this work's context. These techniques will be introduced and explained in detail in the following paragraphs.

The two major types of techniques explored are:

- Color Detection Techniques: where the detection algorithm is based on recognizing objects by detecting and differentiating between the colors of the pixels that form the image of said objects.
- Edge Detection Techniques: where the detection algorithm is based on detecting the edges of objects in order to identify them.

Color Detection :

Binarization

Binarization is a well-known image processing technique that consists of converting a image to a binary image. In image processing, an image is a matrix, and in a binary image the only values we can find are 1 and 0, i.e. 0 for black and 1 for white. We considered using binarization as it helps getting rid of unnecessary information since it reduces the image values to only ones and zeros. However, a straightforward implementation of binarization is not possible since it will differentiate between light and dark colored cars. Nonetheless, a possible implementation exits and needs the usage of two thresholds.

1. 1st Binarization with threshold A

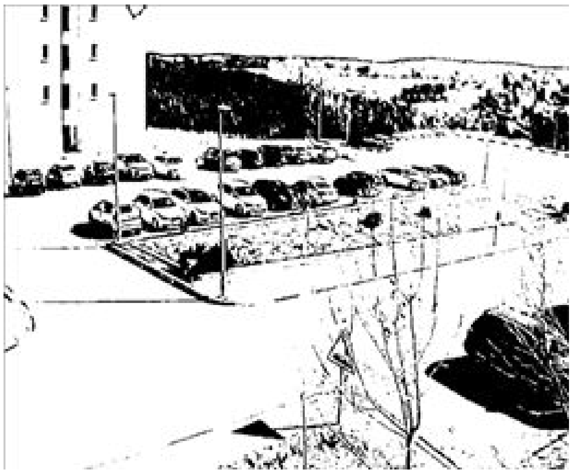

Fig. 8 Binarization with threshold A

The white cars take the value 1, which means that the spot is taken but the black cars remains as 0 which is wrong for now.

2. 2nd Binarization with threshold B

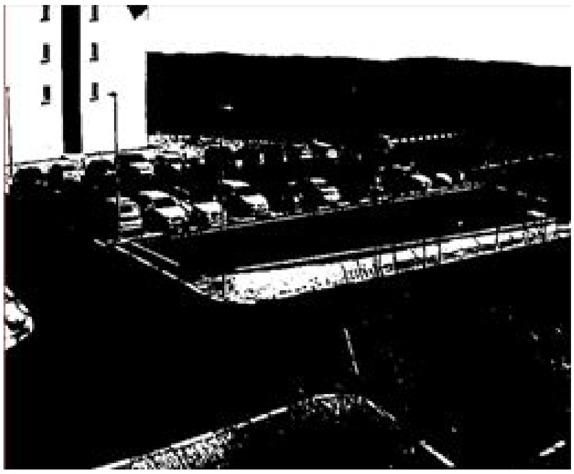

Fig. 9 Binarization with threshold B

The white cars are still detected as 1 and the black cars as 0, however the only change that occurred when using a different a threshold is that the free spots are now 0 and not 1.

3. The detection process

In both image the full spots kept their values in both binarizations however, the free spots have different values in binarization A and B. This means that to detect the free spots, we will only need to compare the two binarization results, when different values are found, this means that the spot in question is free.

This method is very accurate in perfect conditions, but for an outdoor parking this method will be heavily disturbed by change in lighting, shadows and extreme weather conditions. An example is shown below.

The big black shape in the middle is a shadow and this will clearly lead to an invalid detection.

Color Detection Using HSV

This technique can be used to detect and isolate objects of a certain color or whose colors belong to a certain range of colors. Many models are used to represent colors[4]. The HSV model represents colors based on Hue, Saturation and Value and was chosen to be tested due to its low sensitivity to shadows and effects of harsh illumination[5].

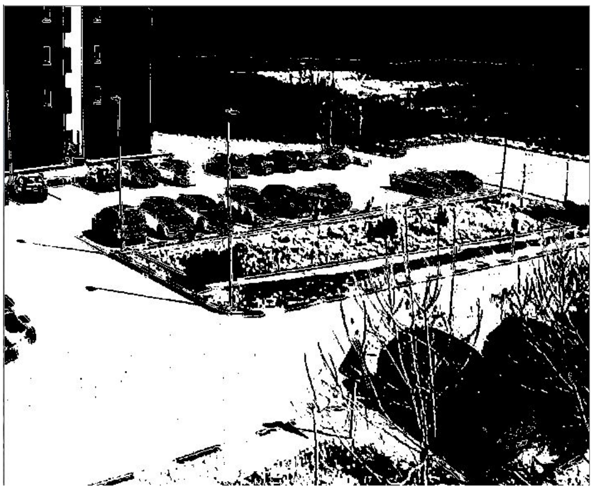

Fig. 10 Color Detection with HSV

In this example, cars with a paint color belonging to a specific color range are identified and represented by the black color in the image. However the drawbacks are that it is hard to find the most optimal color range due to the multitude of car paint colors available. This means that cars with a certain paint color might not be detected.

Edge Detection :

Sobel

The sobel operator is based on convolving the image by using two kernels to approximate the first order derivatives of the changes in both axes of the image [6].

Horizontal

-1	0	1
-2	0	2
-1	0	1

Vertical

-1	-2	-1
0	0	0
-1	-2	-1

Fig. 11 Kernels used by Sobel [7]

This techniques yields the following results:

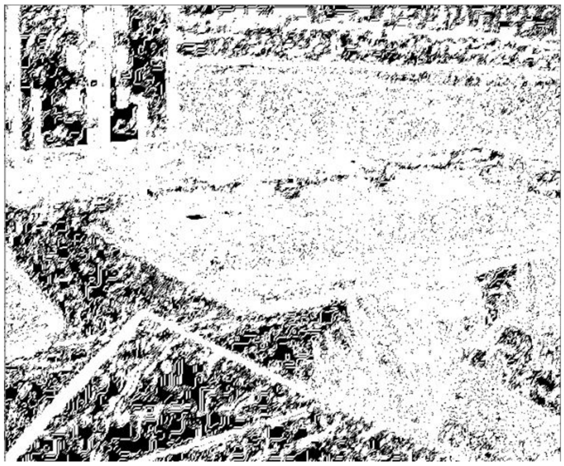

Fig. 12 Result of Sobel

This is the result of Sobel applied on both x and y axes and combined to form the above image (figure 10). Sobel can yield great results in some cases, however it can suffer from inaccurate edge detection and sensitivity to noise as can be noticed in this example.

Laplacian

The Laplacian operator uses the same concept as Sobel in the sense it is based on convolving the image. However, Laplacian only uses one kernel and is based on approximations of the second order derivatives [7].

The laplacian operator

0	-1	0
-1	4	-1
0	-1	0

The laplacian operator (include diagonals)

-1	-1	-1
-1	8	-1
-1	-1	-1

Fig. 13 Kernel used by Laplacian [7]

This is why Laplacian is extremely sensitive to noise and is recommended to be combined with noise reduction in order to produce accurate results [7].

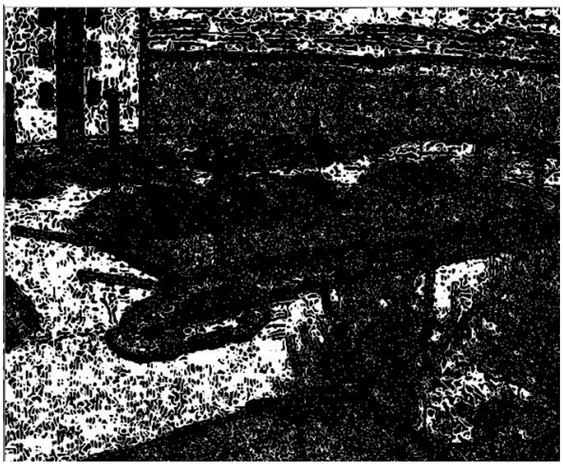

Fig. 14 Result of Laplacian

On the other hand, Laplacian yields sharper and more accurate results when compared to Sobel. This is illustrated by the example above in figure 12.

Canny

Canny is an algorithm that was developed as a multi step process for edge detection [8]. It is based on approximately 4 steps that are:

1. Applying a **Gaussian filter** to the image to reduce noise.
2. Find the **intensity gradients** of the image.
3. **Non-maximum Suppression** which removes any unwanted pixels on the edges of the image to get sharper and thinner edges.
4. **Hysteresis Thresholding** by selecting 2 threshold values (min and max) then any edges above the max value are considered to be "sure edges" and the ones that are above the minimum value that are connected to the "sure edges" are also considered "sure edges" and if they are not connected to any "sure edge" they are discarded[8].

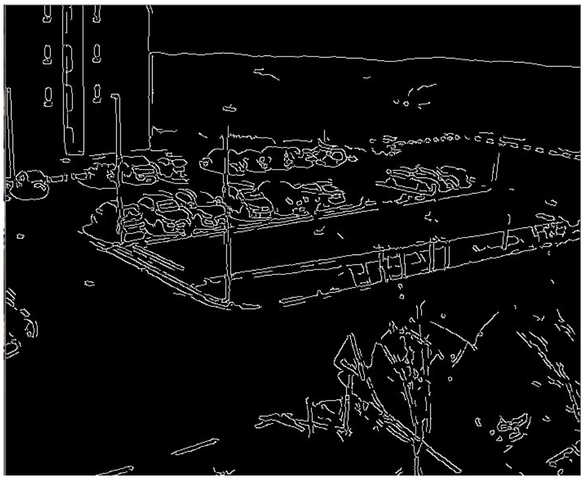

Fig. 15 Result of Canny

Canny produced the best results in terms of accuracy and sharpness in detecting the edges of vehicles as illustrated by figure 14.

Laplacian + Canny

This is the approach of choice for detecting the occupancy of a parking spot in this work and this is the technique that was implemented and is currently used by the algorithm.

Both Canny and Laplacian are applied on 2 different independent copies of the ROI then 2 different means are calculated of the 2 resulting matrices, then the first parking spot status is saved as a boolean value by comparing the resulting mean of using Canny with a predetermined threshold, then the same thing is done for the second parking spot status by using Laplacian.

The 2 boolean results are then evaluated using the OR operator which gives us the real occupancy status of the parking spot.

This means that the parking spot is only empty if using Canny and Laplacian both result in an empty spot.

Canny gives better result when detecting empty spots, however laplacian is better when detecting occupied spots.

The OR operator was used instead of the AND because both techniques are already very accurate by themselves, so if using one technique result in an occupied spot then than spot is likely to be occupied.

Comparative Study

TABLE I

Accuracy Comparison of the different edge detection techniques

Technique Used	Image Without Shadow	Image With Shadow	Image with Shadow and Contrast Limited Adaptive Histogram Equalization Applied
Binarization	95%	10%	10%
HSV	80%	25%	10%
Sobel	80%	40%	60%
Canny	90%	60%	90%
Laplacian	90%	65%	90%
Laplacian + Canny	90%	65%	95%

The table above shows the accuracy of all the techniques that were explored for implementing the car detection algorithm. These results are only relative to the context in which the tests were conducted (Al Akhawayn University car park and security camera). This means that techniques that show better results in comparison to other techniques in this table are not necessarily better.

The study was conducted using 100 different moments from 10 different video captures (10 for each) for each parking spot.

- **Binarization:**The second implementation of the binarization has the best accuracy when using an image free of shadows with an accuracy of 95%; However, the accuracy of this techniques fall substantially when dealing with images that contain shadows, even when using CLAHE.
- **HSV (Color Detection):** Using color detection gives the lowest accuracy when dealing with images without shadows but it has a slightly better accuracy with shadows; However, using CLAHE lowers this accuracy because of how the algorithm sensitive it his to color change.
- **Sobel:** Sobel gives the poorest result out of all the edge detection algorithms but is better overall compared to binarization and color detection.
- **Canny:** this edge detection technique has a 90% accuracy of detecting a parking status regardless of the time of the day and the intensity of the shadow.
- **Laplacian:** this techniques gives the same result as using canny but is slightly better when CLAHE is not used.
- **Laplacian + Canny:** By combining the results of using both techniques, the accuracy of the algorithm increases by 5% compared to only using the Laplacian technique.

Results

To summarize, here are the findings of the comparative study:

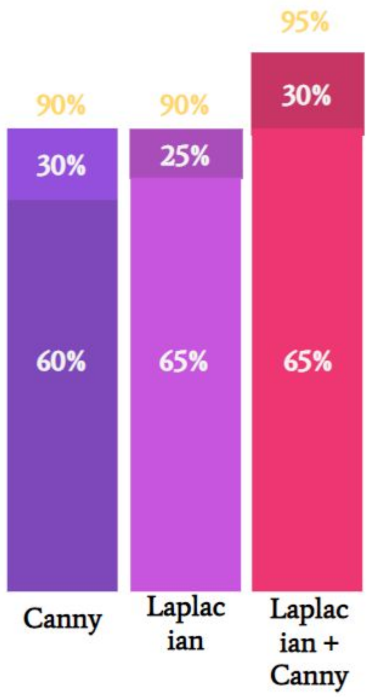

Fig. 16 Accuracy increase using CLAHE

- Using Canny and Laplacian on the same ROI then applying some logic yield the best result.
- The Edge Detection Techniques are the ones that work best in this context.
- Sobel does not work as well as the other edge detection techniques.

- Using Adaptive Histogram Equalization enhances the final result considerably.

Software Demo

ROI are labeled and as we can see, the detection is accurate and only spots 9 and 10 are free. The parking detection is applied only to a part of the parking because the bad camera angle.

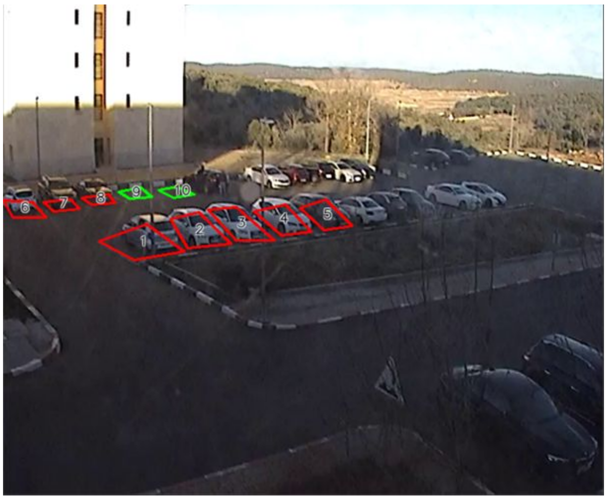

Fig. 17 Sample Execution

Each spot has a database entry, when a spot status is changed, the database gets updated, in this way we do not need to congest the database with unnecessary data requests.

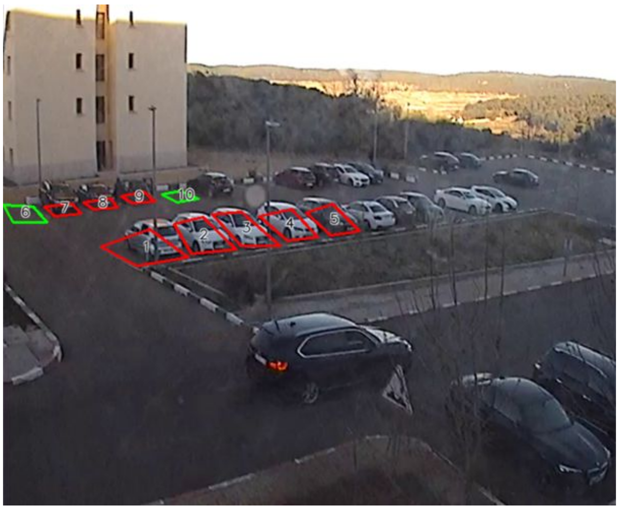

Fig. 18 Sample Execution

Shadows and lighting change do not affect the detection.

Fig. 19 Sample Execution

This service is deployed via a Web Application that was developed using VueJS. The backend was developed using Python.

The web application, shows the parking location, the number of free spots, the weather state(using OpenWeather API),and a timestamp for the last update, it also includes a table with all spots and statuses.

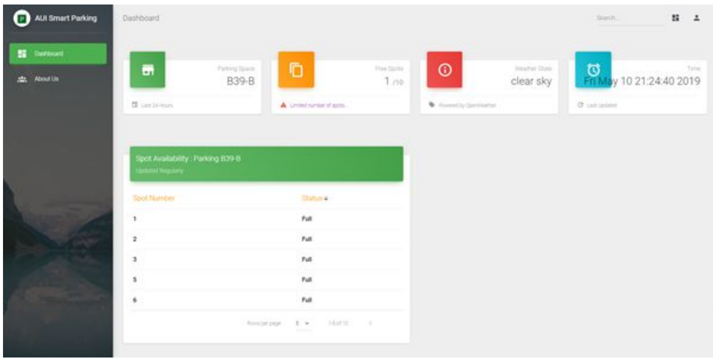

Fig. 20 Web Application Interface

Running Sequence:

1. Database Initialization
2. ROI initialization from a JSON file
3. Get weather State
4. Optimize Image
5. Detection Algorithm
6. Update Database
7. Go to Step 5

Future Scope

The additions that we may consider adding the enhance the overall performance of our program:

- Use better Parking Cameras.
- Detect Illegal parked cars.
- Use Artificial Intelligence for detection.

- Detect parking spots dynamically without using a premade JSON file with polygon indices.

References

[1] Docs.opencv.org. (2019). OpenCV: Color conversions. [online] Available at: https://docs.opencv.org/3.1.0/de/d25/imgproc_color_conversions.html [Accessed 11 May 2019].

[2] Cs.auckland.ac.nz. (2019). [online] Available at: https://www.cs.auckland.ac.nz/courses/compsci373s1c/PatricesLectures/Gaussian%20Filtering_1up.pdf [Accessed 11 May 2019].

[3] Homepages.inf.ed.ac.uk. (2019). Spatial Filters - Gaussian Smoothing. [online] Available at: https://homepages.inf.ed.ac.uk/rbf/HIPR2/gsmooth.htm [Accessed 11 May 2019].

[4] J. A. Moka, "Image Classification With HSV Color Model Processing," Data Science Central, 16-Oct-2017. [Online]. Available: https://www.datasciencecentral.com/profiles/blogs/image-classification-with-hsv-color-model-processing. [Accessed: 11-May-2019].

[5] Hdioud, Boutaina & Mohammed, El Haj Tirari & Rachid, Oulad haj thami & Faizi, Rdouan. (2018). Detecting and Shadows in the HSV Color Space using Dynamic Thresholds. International Journal of Electrical and Computer Engineering (IJECE). 8. 1513. 10.11591/ijece.v8i3.pp1513-1521.

[6] Sobel, Irwin. (2014). An Isotropic 3x3 Image Gradient Operator. Presentation at Stanford A.I. Project 1968.

[7] U. Sinha, "The Sobel and Laplacian Edge Detectors," AI Shack. [Online].Available:http://www.aishack.in/tutorials/sobel-laplacian-edge-detectors/. [Accessed: 10-May-2019].

[8] Sinha, U. (2019). The Canny Edge Detector: Introduction the edge detector - AI Shack. [online] Aishack.in. Available at: http://www.aishack.in/tutorials/canny-edge-detector/ [Accessed 10 May 2019].

[9] "Histograms - 2: Histogram Equalization," OpenCV. [Online]. Available: https://docs.opencv.org/3.1.0/d5/daf/tutorial_py_histogram_equalization.html. [Accessed: 15-Mar-2019].

[10] González Rafael C., González Rafael C., R. E. Woods, and S. L. Eddins, Digital image processing: using MATLAB. Upper Saddle River, NJ: Prentice Hall, 2004.