Vacant Parking Spot Detection

Michael Jones
Department of Electrical Engineering
Stanford University
Stanford, California 94305
Email: jonesmr@stanford.edu

Lewis Li
Department of Electrical Engineering
Stanford University
Stanford, California 94305
Email: lewisli@stanford.edu

***Abstract*—We present a method for (1) identifying parking spaces within parking structures, and (2) a feature detection algorithm that utilizes the results of the first step to determine which parking spots are empty.**

I. Introduction

A. Motivation

It is widely noted that congestion in parking lots in high density areas is a great source of frustration for drivers. With the proliferation of car mounted GPS units and smart-phones, there has been a recent increase in interest in developing ways for drivers to quickly locate vacant parking spots. Current methods rely either on counters, which can only provide the number, but not location of vacant spots, or on installing a choice of different kinds of sensors placed underneath each parking spot, which are not cost effective and often difficult to retro-fit on older parking structures.

B. Prior and Related Work

Presently, many high traffic parking structures such as those found in airports, sports stadiums and shopping centres utilize vehicle entry and exit counters. While this simple approach indicates the number of parking sports that are vacant on a given level, this cannot provide drivers with any information regarding the specific location of vacant parking structures. A system that could keep track of available spots and push the data to driver smart phones or even the ticket received upon entry to the structure would improve driver experience and reduce congestion.

One such system was implemented by the San Francisco Municipal Transportation Authority, which funded the SFpark project to cover 5,100 metered spaces throughout the city using embedded wireless optical sensors [1]. The sensors provide variable accuracy depending on the operational cost the highest accuracy reported is 92% [2]. The sensors are fine-tuned to detect vehicles within a defined area, and as such require clear demarcations on the street. The cost of installing and operating the sensor metering is approximately an annual cost of $500 per unit [3].

Another approach has been to embed ultrasonic sensors on-board vehicles coupled with GPS, which can provide drive-by sensing of parking spot vacancy [4]. However, the placement of the sensor on a mobile platform introduces additional complexity, and deployment of this technology presents its own challenges.

Other approaches using image processing have relied on predominantly on image segmentation based on color histograms. This has the disadvantage of not being robust towards cars that are of similar color to the background; but also requiring variable thresholds depending on lighting conditions which can be a difficult problem to quantify. [5] Another aspect of existing algorithms is the reliance on using hand labelled parking regions - while this is acceptable for small problems, it would be tedious and unrealistic to implement for large parking structures with a large number of spots.

C. Goals

This project intends to build a system that makes use of existing infrastructure found in parking garages (surveillance cameras) to detect vacant parking spots. The aim here is to provide a more cost effective solution while maintaining comparable detection accuracy. Due to the difficulty of detection under different lighting conditions, parking scenarios have been limited to parking structures where lighting conditions are constant. Furthermore, the project intends to establish a method to automatically detect parking lines rather than relying on human intervention.

II. Algorithm

A. Parking Spot Detection

Identification of individual parking spots is done in a controlled scenario where each parking spot is vacant. This is necessary to properly identify the region of interest in the image that corresponds to each parking spot. The goal of the parking spot detection algorithm is to: (1) exclusively identify the parking line dividers, and (2) use neighboring dividers to record the coordinates corresponding to a parking spot.

1) Parking line divider identification: Parking lines are identified by segmenting the image using 8 steps:

- HSV thresholding: filter for HSV values corresponding to yellow parking line dividers
- Vertical-coordinate centroid: any regions lying in the upper half the the image are eliminated
- Solidity: regions with solidity under a threshold are eliminated
- Area: small and large regions are eliminated
- Eccentricity: regions with eccentricity under a threshold are eliminated

- Regions that lie along the same line and whose centroids are within a threshold value are joined to account for breaks in the parking line divider, as may be the case for faded paint
- Vertical-coordinate centroid: any outlier regions
- Major axis length: regions with major axis length under a threshold are eliminated

2) Parking spot coordinates identification: Once the parking line dividers have been identified, the endpoints of each line are computed. These endpoints and the endpoints of the neighboring parking line divider represent the four vertices of a quadrilateral which corresponds to the base of a parking spot. Four more vertices corresponding to the vertices of a 3-dimensional volume that the car occupies are computed by subtracting a height value from each base vertex coordinate. The height value is computed as a function of the parking line divider major axis length and orientation, such that the height for parking line dividers with a larger orientation is smaller. This results in an 8-coordinate bounding box that roughly defines the area in the image in which a car would reside. In order to account for occlusion by neighboring cars, which may enter into the bounding box of adjacent bounding boxes, five of these coordinates are chosen to form a bounding box which better represents the region a car is visible in the image. Figure 1 presents the algorithm for identifying individual parking spots and 2 illustrates the resulting bounding box chosen for parking spaces with different orientations.

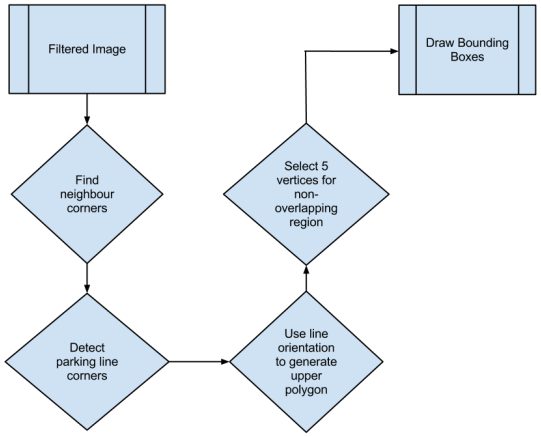

Fig. 1. Overview of how the parking space detection algorithm will work

B. Vehicle Detection

The vehicle detection makes use of both the image of the empty parking lot that was used to generate the bounding boxes and the coordinates of the bounding boxes themselves. Figure 3 gives a high level overview of how the vehicle detection portion of the project works.

1) SIFT Detection: The Scale Invariant Feature Transform developed by Dr. David Lowe in 1999 can be used to detect and describe local features in various image. It involves sampling image gradients in the scale space and then generating

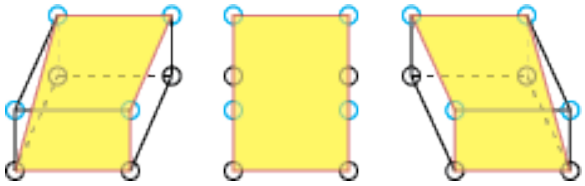

Fig. 2. 5-coordinate bounding box (shaded region). Note that the skewness of the box increases with parking divider angle. This is more indicative of what regions of the vehicle can be seen by the camera if there are occluding cars in adjacent spots.

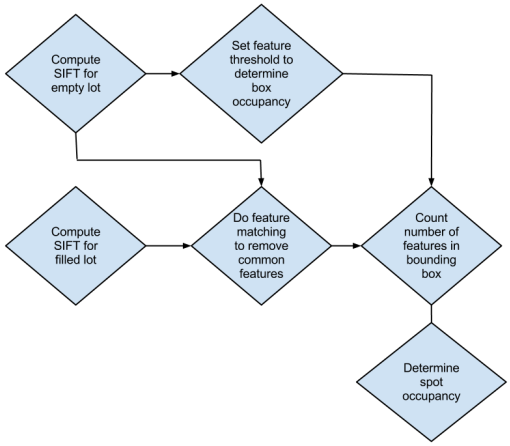

Fig. 3. Overview of how the car detection algorithm will work

an orientation histogram and computing a feature vector, the details of which can be found in [6]. SIFT was designed to be invariant to image translations and rotations, to scale changes and robust against illumination changes. Although in the present scope of the project the images will not be translated or rotated, many security systems maintain cameras on rotary bases to provide extended field of views. SIFT would be versatile enough to be used in such a situation (perhaps in combination with RANSAC). Nevertheless, SIFT proved to detect a relatively greater number of salient features in our test data as compared to SURF or Harris Corners.

2) FLANN: We used SIFT on both the empty space image as well as the image with vehicles. By matching the feature descriptors found in both images and removing them from the second image, we can isolate only the features that are associated with any vehicles in the image. The process we used to remove features in common is Fast Approximate Nearest Neighbours (FLAAN), which was also developed by David Lowe [7]. We matched features based on their descriptors but also based off their (x,y) position. Since the camera has not moved between the two images, matched features should be fairly close in pixel proximity to each other. We found that a threshold of less than 25 pixels worked well for both our test environments.

3) Feature Thresholds: Once we have matched the features from each image to its corresponding image of the empty lot, we are primarily left with the interesting features that

represent the vehicles. At this point, we utilized the bounding box for each spot that was computed in the first portion of this algorithm to compute the number of features that reside inside. Since the majority of the parking spots are at an angle to our camera, we can only look at a subset of the bounding box without inadvertently counting features from neighbouring cars that may be occluding the full parking spot. Instead we choose a polygon comprised of the 5 coordinates – see Figure 2 for the corresponding vertices used for different parking space orientations. After identifying the number of features in each spot, we can judge this number against a threshold to determine whether there is a sufficient number of features in each spot to warrant considering it occupied. We computed this threshold by comparing it to the number of features that were present in the same bounding box on the empty parking lot image. A threshold of 1.5 times the number of features in the empty lot gave us good results. Ties are broken in favour of considering the parking spot occupied.

III. Data Collection

We collected data from Stanford Parking Structure 2 with a tripod mounted DSLR camera. We picked two separate locations where the camera was able to give a view of 6 and 7 parking spots respectively. We then moved two cars into various spots and recorded images of both the stationary cars and driving cars. Finally, we observed test cases where a car was parked between two spots in addition to pedestrians in one of the spots. We first ran the parking spot detection algorithm on the empty images and drew the appropriate bounding boxes. These are shown in Figures 4 and 5.

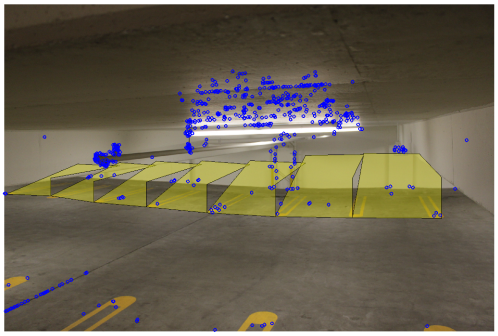

Fig. 4. Empty partking lot 1 with SIFT features shown in blue and bounding boxes in yellow

IV. Results

Detection performance varied depending on the feature count multiplier value which served as a threshold for determining whether a parking spot was vacant or occupied. The greatest accuracy was obtained when the multiplier value was chosen to be 1; however, this also resulted in the greatest number of false positives for Scenario 1. Performance metrics

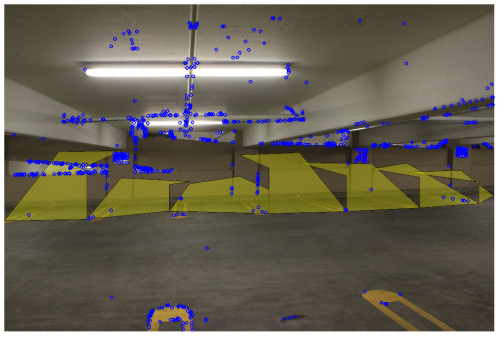

Fig. 5. Empty partking lot 2 with SIFT features shown in blue and bounding boxes in yellow

for this threshold are included in Figure 6. The accuracy, precision, specificity and sensitivity as a function of the feature count multiplier for each scenario is summarized in Figure 7 and Figure 8.

Metric	Scenario 1	Scenario 2	Scenario average
True +	72	58	65
True -	254	166	210
False +	5	8	6.4
False -	5	6	5.5
Accuracy	0.97	0.94	0.96
Precision	0.94	0.88	0.91
Sensitivity	0.94	0.91	0.92
Specificity	0.98	0.95	0.97

Fig. 6. Performance using 1x feature count multiplier threshold

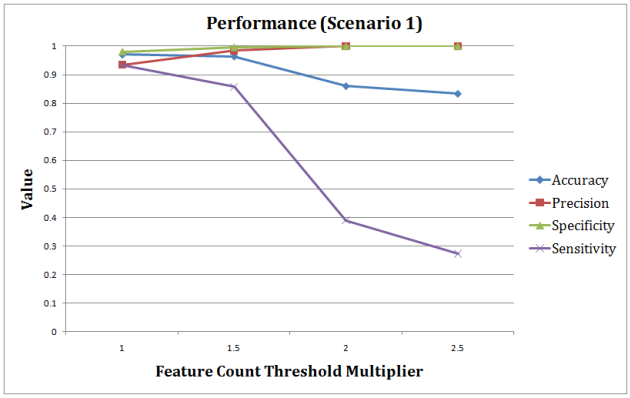

Fig. 7. Performance curves for scenario 1

V. Discussion

The following sections examine results under different scenarios in greater detail.

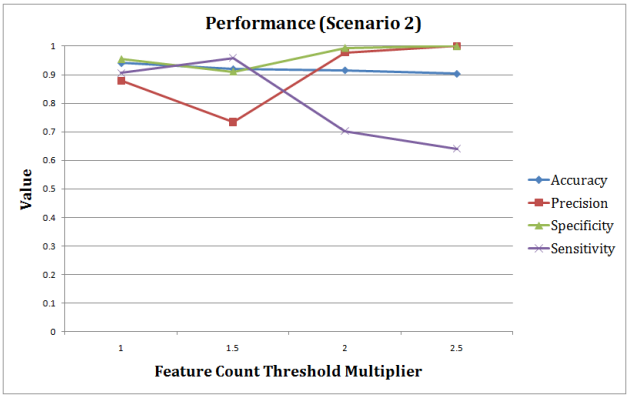

Fig. 8. Performance curves for scenario 2

Fig. 9. Baseline case where vehicles are located next to each other but close to the camera

A. *Parking Spot Detection*

In the first set of test data, the DSLR camera was mounted on a tripod at an elevation that was approximately 10 feet above the parking lot. This is meant to be indicative of an overheard surveillance camera that gives a slightly angled view point of 7 parking spots. We did not consider the two partial spots at the base of the image, since these were filtered during the parking spot identification step. The parking spots furthest to the right are viewed straight on and give well defined bounding boxes as shown in yellow in Figure 4. As we examine parking spots towards the left, the bounding boxes gradually get smaller and stretched as any neighbouring cars would occlude more of the region. One can expect that the left spots will be harder to detect than the right – this will be discussed in a later section.

The second scenario is shown in Figure 5, which depicts the image from a camera that is located 5 feet above the ground and is more indicative of a wall mounted camera. Here we have a view of 6 parking spots. In this particular case, the endpoints of the divider between the second and third spot were incorrectly calculated, and thus resulted in skewed bounding boxes for the respective parking spaces. This can be attributed to the fact that the parking line directly faces the camera, and thus the bottom of the device corresponding to the closest point appears to be larger. Moreover, the spot is slightly oriented towards the right, resulting in the bottom left corner being selected as the endpoint and the top right as the other endpoint. This causes the resulting mask to be much wider than it should be.

B. *Successful Vehicle Detection*

The following cases illustrate robustness in the algorithm to certain conditions.

1) Baseline cases: In Figure 9, two cars occupy spots 1 and 4, respectively. This is considered a baseline case where the only populated spots are those closest to the camera. In this case, even adjacent cars result in little to no occlusion. We see that despite the sub-optimal bounding box generated by the parking spot detection algorithm in this case, the detection was still able to successfully identify the spot as being occupied.

2) High Angle Viewpoints: Here we consider Figure 10, which depicts two adjacent vehicles located in spots 4 and 5 of the second scene. We see that the bounding boxes do not encompass any neighbouring regions. There is a slight overlap of the red car into spot 6 but this is not sufficient to register as occupied. In this scenario parking spots 6 and 7 are occluded by the overhead beam and were not fully visible, which made detecting cars in them challenging. One method to judge whether a bounding box is acceptable is to measure the distance between the vertex which corresponds to the concave portion of the bounding box polygon and its opposite side – when this distance is small, we may judge the parking spot is too far from the camera for the algorithm to reliably determine if it is occupied.

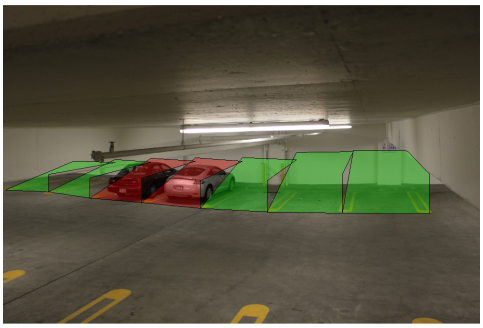

Fig. 10. Far away case

3) Partial Occlusion: Figure 11 shows an example of another vehicle driving across the scene. We note that the passing vehicle does not generate enough features to generate any false positives. In reality, this could be fortified by reporting the occupancy of a spot by averaging occupancy results over several frames, as it is unlikely a car will be parked in the middle of the lane.

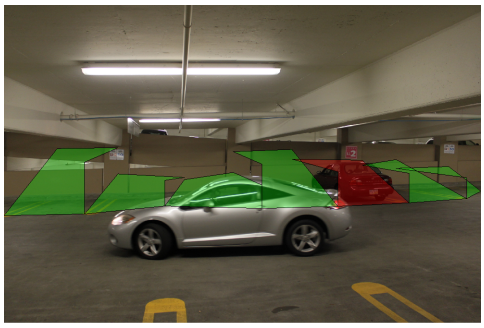

Fig. 11. Partial occlusion caused by passing cars

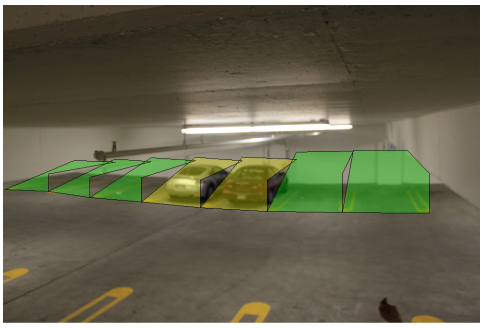

Fig. 13. Out of focus image

4) Partially Parked Car: Figure 12 shows a similar case where a car is only partially parked. This could be result of a parking backing out to leave or a botched parking job. In this particular example, the car is sufficiently in the spot enough that our algorithm registers the spot as being occupied. Likewise with the partial occlusion case, averaging occupancy detection over a few seconds would allow the system to be more robust towards drivers who are adjusting their parking.

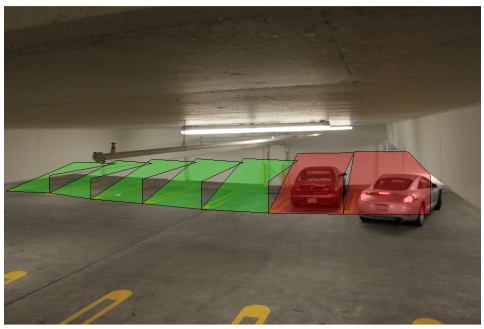

Fig. 12. Partially parked car case

C. Unsuccessful Vehicle Detection

While collecting data we also tried to find cases in which our detection algorithm would fail, which are included below.

1) Out of Focus: The first case (Figure 13), involved taking an image that was out of focus. This causes the SIFT detector to pick up very few features, and thus not meeting the thresholds required to mark a spot as occupied. In reality, a surveillance camera would generally not be out of focus, and autofocusing algorithms are well studied. However, this does suggest that SIFT, at least in our particular scenario, may not be blur invariant.

2) Incorrectly parked car: Figure 14 shows an example of where a driver has parked in the middle of two parking spaces. In this particular case, only the space on the right was detected to be occupied while the space on the left predicted to be vacant. This can be attributed to the nature of how we selected our bounding boxes – the region which the car resides in the left spot as seen by the camera would be partially occluded if the car was correctly parked in the space on the right. This example illustrates one of the issues with how we are detecting cars. A suggested workaround would be to make the detection a multi-step approach, where we first search the bottom four coordinates of the parking space and based on the result, subsequently decide whether to search the other faces of the box or the 5-coordinate bounding box.

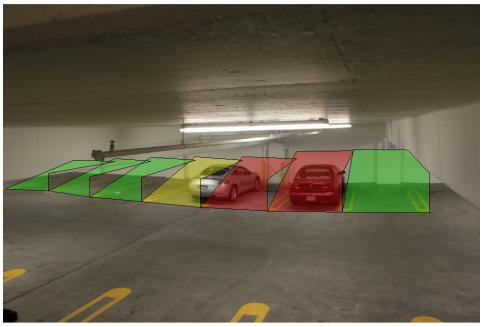

Fig. 14. Incorrectly parked car

3) Pedestrian: We also considered the case where a pedestrian was standing in a parking spot. Figure 15 depicts an example where the pedestrian was not detected in the parking spot. The bounding box at that particular box happens to be skewed and the person is standing mostly in the region where a neighbouring car would expect to occlude. Once again, a multi-step approach as suggested in the previous section may help detect the pedestrian. Conversely, in Figure 16, the individual is standing in the area that would not be occluded by a neighbouring car resulting in a positive occupancy reading.

In reality, although it is unlikely a pedestrian would stand in the middle of a parking spot for any lengthy period of time, we would still like to detect the parking spot as being occupied if this were the case – this can be accomplished by averaging over several frames.

Fig. 15. Pedestrian resulting in a negative occupancy reading

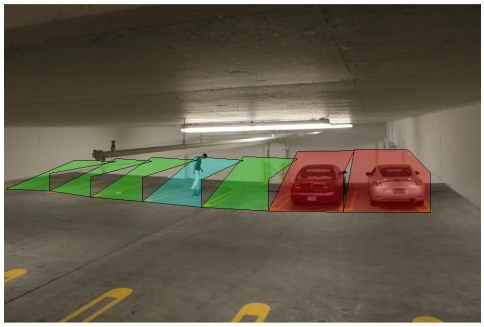

Fig. 16. Pedestrian recsulting in a positive occupancy reading

VI. Future Work

This current iteration of the algorithm used controlled conditions in indoor parking structures, which has the benefit of consistent parking space dimensions and lightning. To make such a system universal, it should be implementable in outdoor parking lots subject to varying illumination, and across a variety of parking line divider shapes and sizes. A possible extension could be to use the Hough transform to detect linear dividers and use it in conjunction with the currently implemented brightness map to generate the parking spot location. To be more robust towards lightning, it would be ideal to move away from a color threshold technique and consider a machine learning method. For instance, a large amount of data could be collected identifying where drivers are parking in a particular region. Expectation maximization could be then be applied to detect clusters which would correspond to parking spots.

Another aspect of the algorithm that could be improved would be the threshold for the number of features that would classify a spot as being occupied or vacant. The present method computes it as a multiple of the number of features in each spot in the image of the empty lot. It would be desirable to use a backpropagation neural network to determine what the threshold should be for each spot. This would require a lot of training data and hand labeling each spot in the training data as being occupied or vacant.

To transform this algorithm into a production level implementation, it would be necessary to quantity the relationship between camera field of view, height, and other camera properties with the number of spots that it can accurately detect. This would also require investigation into the relationship between the spot orientation and the distance from the camera. Machine learning techniques such as Support Vector Machines could be used to determine whether the occupancy of a spot could be accurately determined given the spot's position and orientation and the camera parameters. This could be used to compute the optimal placement of cameras within a given parking structure. Furthermore, adjacent cameras could offer views of the same parking spot from different viewpoints with different occlusion properties. The currently implemented SIFT descriptors and matching could be used to identify which spots are visible to multiple cameras, and provide additional robustness. Another situation that the SIFT features can be used is when the camera is mounted on a rotating base. SIFT and RANSAC can be used in conjunction to provide detection over a higher field of view for each camera.

A further improvement could be made by generating a database of vehicle parts and their corresponding SIFT descriptors through test data and hand labeling. Thereafter, each detected descriptor could be compared to this database to distinguish between vehicle or a pedestrian or stationary objects. This would replace the current feature matching step.

Finally, the results of the algorithm should be pushed in real-time to smart phones, which would require a mobile app and for the detection to run much faster. Currently, the bottleneck is the SIFT detection which is implemented using OpenCV. There are GPU-accelerated versions of SIFT available, one of which the co-author has developed that can run SIFT 50 times faster than a serial implementation [8].

Appendix

Michael Jones worked on the parking spot detection algorithm. Lewis Li worked on the vehicle detection algorithm. Both worked on the poster and report.

Acknowledgements

The authors would like to thank Maryam Daneshi for her input on the project and Prof. Bernd Girod for his EE 368 course. Furthermore, the authors would like to acknowledge Michael Xu and Nathan Keegan for lending their cars to collect test data.

REFERENCES

[1] R. Gordon, *High-tech parking meteres premiere in S.F.*, SFGate. July 27, 2010. http://www.sfgate.com/cgi-bin/article.cgi?f=/c/a/2010/07/27/BA2P1EK39G.DTL

[2] *Parking sensor performance standards and measurement*. SFPark. July 18, 2011. http://sfpark.org/wp-content/uploads/2011/09/SFpark_SensorPerformance_v01.pdf

[3] E. Jonietz, *Finding a Parking Space Could Soon Get Easier*. Technology Review. February 8, 2010. http://www.technologyreview.com/printer_friendly_article.aspx?id=24497

[4] S. Mathur, T. Jin, N. Kasturirangan, J. Chandrashekharan, W. Xue, M. Gruteser, W. Trappe, *ParkNet: Drive-by Sensing of Road-Side Parking Statistics*. WINLAB, Rutgers University. http://www.winlab.rutgers.edu/~suhas/SuhasMathur_Mobisys2010.pdf

[5] Nicolas True, *Vacant Parking Space Detection in Static Images*,University of San Diego, San Diego USA, 2007

[6] David G. Lowe *Distinctive image features from scale-invariant keypoints*. International Journal of Computer Vision Issue 60, 2004

[7] Marius Muja and David G. Lowe, *Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration*, in International Conference on Computer Vision Theory and Applications (VISAPP'09), 2009

[8] Lewis D. Li *Technical Report: GPU Accelerated SIFT3 Implementation*. MDA Space Missions, Brampton, Ontario Canada, 2010