Self-Supervised Aerial Image Analysis for Extracting Parking Lot Structure

Young-Woo Seo, Nathan Ratliff and Chris Urmson
Robotics Institute
Carnegie Mellon University
{ young-woo.seo, ndr, curmson}@ri.cmu.edu

Abstract

Road network information simplifies autonomous driving by providing strong priors about environments. It informs a robotic vehicle with where it can drive, models of what can be expected, and contextual cues that influence driving behaviors. Currently, however, road network information is manually generated using a combination of GPS survey and aerial imagery. These manual techniques are labor intensive and error prone. To fully exploit the benefits of digital imagery, these processes should be automated. As a step toward this goal, we present an algorithm that extracts the structure of a parking lot visible from a given aerial image. To minimize human intervention in the use of aerial imagery, we devise a self-supervised learning algorithm that automatically generates a set of parking spot templates to learn the appearance of a parking lot and estimates the structure of the parking lot from the learned model. The data set extracted from a single image alone is too small to sufficiently learn an accurate parking spot model. However, strong priors trained using large data sets collected across multiple images dramatically improve performance. Our self-supervised approach outperforms the prior alone by adapting the distribution of examples toward that found in the current image. A thorough empirical analysis compares leading state-of-the-art learning techniques on this problem.

1 Introduction

The 2007 DARPA Urban Challenge demonstrated the potential for driverless automobiles to operate in urban daily life in the near future [Urmson *et al.*, 2008]. Important to the success of the Urban Challenge was the availability of detailed digital road network information. Road network information simplifies autonomous driving by providing strong priors about driving environments. That is, it tells a robotic vehicle where it can drive, models of what can be expected and provides contextual cues that influence the driving behavior. For example, road network information allows a robotic vehicle to anticipate upcoming intersections (e.g., that the intersection is a four-way stop or that the robot must conform to precedence rules) and other fixed rules of the road (e.g., speed limits). However, existing road network information is manually generated using a combination of GPS survey and aerial imagery. These techniques for converting digital imagery into road network information are labor intensive, reducing the benefit provided by digital maps. To fully exploit the benefits of digital imagery, these processes should be automated. In this paper, as a step toward automatic generation of the road network information from aerial images, we present a self-supervised learning algorithm that learns appearances of parking lots from self-generated examples and uses these examples to extract their structures.

Automatically extracting parking lot structures is challenging because their shapes are only approximately regular and aerial images are noisy. Specifically, the shapes of parking lots in images look similar, but they are slightly different. Thus, the structure of each individual parking lot needs to be analyzed separately. Noise in the images makes parking spots inconsistent in appearance due to vehicle occupancy, occlusions by other structures such as trees and adjacent buildings, or differing illuminations (e.g., under the shade of buildings.)

In order to handle these problems effectively, we propose a hierarchical approach to generating and filtering candidate hypotheses. Variations in illumination, occlusions, and parking spot geometry dictate the use of machine learning approaches trained on examples extracted from the image of interest. Our approach is implemented as two layers, a low-level layer, which extracts and compiles geometrical meta-information for easy-to-find parking spots, is highly accurate and serves as a prime source of examples for self-supervised training. The high-level layer uses outputs from the low-level layer to predict plausible candidate hypotheses for more difficult parking spot locations and then filters these hypotheses using self-trained learners. Our method is described in detail in Section 3.

2 Related work

The framework of self-supervised learning has recently been attracting attention from the robot learning community, since it requires no (or substantially less) human involvement for carrying out learning tasks. This framework is highly desirable for robot learning because it is usually hard to collect large quantities of high-quality human-labeled data from any real world robotic application domain. Self-supervised learning frameworks typically utilize the most precise data source to label other data sources that are complementary, but unlabeled.

For example, conventional range finders provide accurate distance estimates between a robot and surrounding objects, but the range is limited. Sofman and his colleagues use those local range estimates as self-labeled examples to learn relations between the characteristics of local terrain and the corresponding regions in the aerial images [Sofman *et al.*, 2006]. These learned relations were used to map aerial images to long range estimates of traversability over regions that the robot is going to explore. Similarily, Stavens and Thrun utilize laser range measurements to predict terrain roughness [Stavens and Thrun, 2006]. They first analyze the associations between inertial data and laser readings on the same terrain and use the learned rules to predict possible high shock areas of upcoming terrains. Lieb and colleagues devised a self-supervised approach to road following that analyzes image characteristics of previously traversed roads and extracts templates for detecting boundaries of upcoming roads [Lieb *et al.*, 2005].

Our algorithm fits this self-supervised framework well. A low-level analysis phase extracts lines forming parking lot lane markings, resulting in a collection of canonical parking spot image patches which can be used as training examples. We additionally use these initial parking spots to guide a random selection of negative examples.

Most prior work in parking lot analysis [Wu *et al.*, 2007; Fabian, 2008; Huang *et al.*, 2008] focused primarily on detecting empty parking spots in surveillance footage when the overall geometrical structure of the parking lot is known. Our work addresses the more general problem of extracting the parking lot structure from a single overhead image.

To the best of our knowledge, automatic generation of road network information from aerial imagery has not been extensively investigated yet. The most similar work to ours is Wang and Hanson's that uses multiple aerial images to extract the structure of a parking lot for simulation and visualization of parking lot activities [Wang and Hanson, 1998]. Multiple images at different angles are used to build 2.5 dimensional elevation map of the parking lot. This usage of multiple images makes it difficult to generalize their method because it is not easy to obtain such images on the same geographic location from publicly available imagery.

3 Method

The structure of a parking lot in an aerial image is characterized by the layout of a set of parking blocks and their parking spots. We define the extraction of parking lot structure as the detection of all visible parking spots.[1] Figure 1 illustrates how a parking lot is represented in this paper. Our algorithm parameterizes each individual *parking spot* by its height, width, orientation, and centroid location in image coordinates. We define a *parking block* as a row of parking

[1]In particular, we ignore entirely occluded parking spots such as spots blocked by trees.

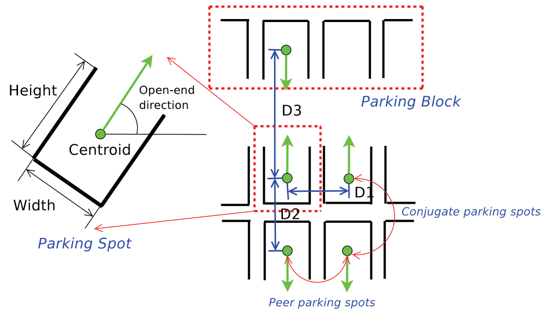

Figure 1: This illustration depicts the parking spot and parking block representations used throughout this work.

spots all oriented in the same direction. Each parking block is characterized by the distance between neighboring parking spots in the block (i.e., "$D1$" in figure 1). Parking blocks are related to each other by two distance measures: the distance between conjugate parking spots (i.e., "$D2$") and the distance between blocks (i.e., "$D3$" in the figure 1).

If the image locations of all visible parking spots are known, it would be trivial to estimate alignment and block parameters. However, in practice we must estimate these parameters from a given image to determine the parking lot structure. In what follows, we describe in detail our hierarchical approach to detecting parking spots. Section 3.1 presents the image processing steps involved in the low-level image analysis layer. This layer accurately extracts a set of easily found parking spots from the image. Section 3.2 details the high-level processing layer which then extrapolates and interpolates the spots found by the low-level analysis to hypothesize the locations of the remaining parking spots. We then discuss our self-supervised hypothesis filtering approach, which removes erroneous hypotheses from the collection.

3.1 Low-Level Analysis: Detecting Canonical Parking Spots

Geometrical and image characteristics differ between parking lots. Most overhead satellite parking lot images contain a number of well-illuminated empty parking spots. Our low-level analysis extracts these easy-to-find spots to be used by the high-level analysis as "seeds" for additional hypothesis generation and by the final filtering stage as canonical self-supervised training examples to adapt the filter to this particular image. The low-level layer carries out multiple image processing steps: line extraction, line clustering, and (parking) block prediction.

Straight lines are important to understanding the shape of a parking lot. We extract lines using the approach proposed by [Kahn *et al.*, 1990]. The approach computes image derivatives to obtain intensity gradients at each pixel and quantizes the gradient directions using predefined ranges. A connected component algorithm is then used to group pixels assigned the same direction to form line supporting regions. The first principal eigenvector of a line supporting region determines the direction of the line.

Although a majority of extracted lines may align with lane markings of the parking lot, some of them come from other

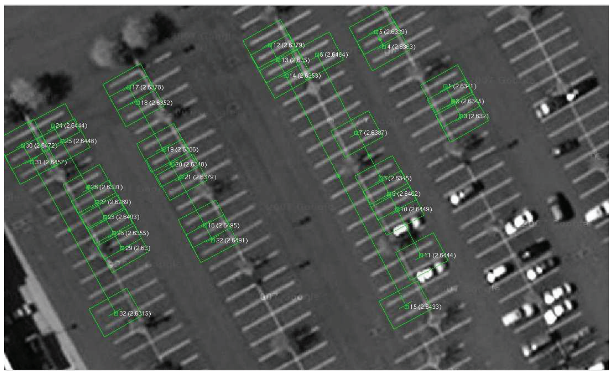

Figure 2: An illustrative example image is shown. The low-level analysis produces the estimated parameters and a set of canonical parking spots templates that are depicted by (green) rectangular patches around their centroids. There are 55 canonical parking spots obtained in this example. Viewed best in color.

image regions such as road lanes or contours of adjacent buildings. Since we only want the lines aligned with the lane-markings of the parking lot, it is necessary to remove lines that do not belong to the parking lot structure. To this end, we group the extracted lines into clusters based on their orientations and remove lines that are either too short or too long from the cluster with the largest member.

For the parameter estimation, we first estimate the nominal height of a parking spot by computing the mode of each line in the selected cluster. We next build a Euclidean distance matrix across all possible line pairs, quantize the distances and compute the mode to obtain the width and height of parking spots within a lot. Finally, we quantize the orientations of lines and compute the mode again to estimate the orientation of parking spots' open-end.

The completion of these image processing steps results in generating few, highly accurate, initial estimates of true parking spots. Figure 2 shows rectangular patches around the image coordinates of parking spots obtained using this low-level analysis.

This low-level analysis is then extended to additionally identify entire parking blocks. To this end, we project the centroids of all the inital parking spots onto a virtual line whose orientation is the mean of the initial parking spots' orientation. This projection returns distances of centroids from the origin, $\rho_i = c_{i,x}\cos(\theta_i) + c_{i,y}\sin(\theta_i)$, where $c_{i,x}$ and $c_{i,y}$ are image coordinates of parking spot centroid and θ_i is the open-end orientation of the ith parking spot. After projection, boundaries between parking blocks are clearly visible and the distance between peer parking spots (i.e., $D1$ in the Figure 1) is used to determine boundaries between parking blocks. From the discovered parking blocks, we finish the parameter estimation by computing three distances between parking blocks (i.e., $D1$, $D2$, and $D3$ in the Figure 1).

3.2 High-Level Analysis: Interpolation, Extrapolation, Block Prediction, and Filtering

The high-level layer is intended to detect all the visible parking spots in an image. To this end, it first hypothesizes parking spot locations based on the parameters estimated by the low-level layer. It then filters these hypotheses by classifying the rectangular image patches around these hypotheses using self-supervised classifiers.

Parking Spot Interpolation and Extrapolation

A parking spot hypothesis represents an image location of the centroid of a potential parking spot. A rectangular image patch around the hypothesis is evaluated to determine if local characteristics of the image are similar to that of a true parking spot. To cover the set of image regions that possibly contain true parking spots but are initially uncovered, we use the image coordinates of centroids of the initial parking spots as the starting points in each of the discovered parking blocks. The image coordinates used for this hypothesis generation are located at the both end of an estimated parking block. We then generate parking spot hypotheses by selecting image locations through three processes: interpolation, extrapolation, and block prediction. The interpolation procedure chooses image coordinates between two end parking spots in a parking block, whereas the extrapolation procedure extends hypotheses beyond two boundary parking spots. The estimated width of a parking spot is used as the spatial interval between parking spot hypotheses. Finally, block prediction aims at discovering the missing parking blocks. We use the estimated parking block distances and select image regions to test existence of parking blocks.

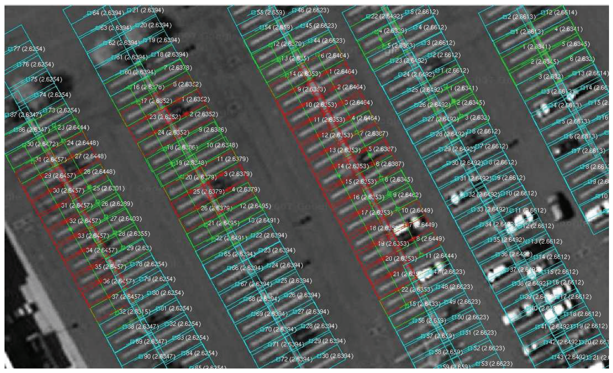

Figure 3: A set of the generated parking spot hypotheses is depicted by rectangular patches. Different colors indicate results of different hypothesis generation processes (red patches by the interpolation, cyan ones by extrapolation, and green ones by the low-level analysis). In this example image, there are 186 true parking spots and 205 parking spot hypotheses. Viewed best in color.

Self-supervised Hypothesis Filtering

The hypothesis generation process produces n parking spot hypotheses represented by the corresponding number of image patches, $g_1, ..., g_n$. Figure 3 shows a complete set of the generated parking spot hypotheses. Each of the parking spot hypotheses is evaluated to determine if it is a parking spot. We formulate this decision problem as binary classification for assigning a label, $y_i \in \{-1, +1\}$, to a given patch vector, $\mathbf{g}_i$, which is an $m = \text{height} \times \text{width}$-dimensional column vector. The individual components in this vector are intensity values of a grayscale image patch. Our experiments compare three state-of-the-art learning techniques for this binary classification task: Support Vector Machines (SVMs), Eigenspots, and Markov Random Fields (MRFs).

Support Vector Machine. SVMs are a *de facto* supervised learning algorithm for binary classification. They seek to find the hyperplane that maximizes a notion of margin between each class [Bishop, 2006]. Linear SVMs are fast, have publicly available implementations, and handle high-dimensional feature spaces well.

Eigenspots. Since processing high-dimensional image patches is computationally expensive, we reduce the dimension of our feature space by using principal component analysis (PCA). This is intended to find the principal subspace of the self-supervised parking spots obtained by the low-level analysis; we retain the top $k \ll m$ dimensions of the space. In homage to Turk and Pentland [Turk and Pentland, 1991], we call the eigenvectors of the parking spot space extracted by this method the "Eigenspots" of the space.

We use this new space in two ways. Let p be the number of positive examples extracted by the low-level analysis, and denote by $\mathbf{\Psi} = \frac{1}{p}\sum_i \mathbf{g_i}$ the average canonical parking spot. Our first technique simply measures the distance from a candidate patch to the center of the space (i.e., the mean canonical parking spot) using the following eigen-weighted distance measure. Given a new image patch $\mathbf{g}$, we compute

$$T(\mathbf{g}) = \|\mathbf{D}^{-1/2}\mathbf{E}^T(\mathbf{g} - \mathbf{\Psi})\|, \tag{1}$$

where $\mathbf{D}$ is a diagonal matrix containing eigenvalues $\lambda_1, \ldots \lambda_m$ of the covariance matrix used in the PCA computation, and $\mathbf{E}$ is a matrix whose columns are the eigenvectors. $T(\mathbf{g})$ is known as the Mahalanobis distance [Bishop, 2006] from the origin of the Eigenspot space. If this distance is less than a threshold, we classify the new image patch as a parking spot.

Our second technique simply pushes the examples through the PCA transformation before training the SVM classifier. Specifically, we transform each example as $\widetilde{\mathbf{g}} = \mathbf{D}^{-1/2}\mathbf{E}^T(\mathbf{g} - \mathbf{\Psi})$.

Pairwise Markov Random Fields. SVMs and Eigenspots only consider the local characteristics of an image patch to perform the binary classification. Since their performance is limited by the distribution of the training data, it is useful to investigate neighboring image patches around the patch of interest as well as to look at the local characteristics of the image patch. To implement this idea, we use a pairwise Markov Random Fields (MRFs) [Li, 2000]. A pairwise MRF $\mathcal{H}$ is an undirected graphical model that factorizes the underlying joint probability distribution $P(G)$ by a set of pairwise cliques. [2] An undirected graph, $\mathcal{H}$, is comprised of a set of nodes and their edges, where a node represents a random variable and an edge between nodes represents dependence between them.

[2]There may be bigger cliques in the graph, but the pairwise MRF only consider pairwise cliques.

In this work, there are two different types of nodes: observed and unobserved nodes. An observed node corresponds to an image patch whereas an unobserved node is the true label of the observed node. Although we observe the value of a node ($\mathbf{G}_k = \mathbf{g}_k$), the true label of the node ($Y_k = y_k \in \{-1, +1\}$) is not observed. The task is then to compute the most likely values of unobserved nodes, Y, given the structure of the undirected graph, $\mathcal{H}$, and charactersics of image patches, G. The joint probability distribution is modeled as

$$P(G) = \frac{1}{Z}\prod_{i=1}^{N}\Phi(G_i, Y_i)\prod_{j \in N(i)}\Psi(Y_i, Y_j)$$

where $\Phi(G_i, Y_i)$ is a node potential, $\Psi(Y_i, Y_j)$ is an edge potential, Z is the partition function, and $N(i)$ is the set of nodes in the neighborhood of the ith node. In our work, we only consider first-order neighbors.

We estimate the node potentials using a Gaussian Mixture model (GMM) [Bishop, 2006]; we assume that candidate parking spots are generated from a mixture of multivariate Gaussian distributions. Since we have two labels, each node has two potentials: a potential being a parking spot, $\Phi(G_i, Y_{j=+1})$ and the other potential being not a parking spot, $\Phi(G_i, Y_{j=-1})$. The edge potential is computed by the Potts model [Li, 2000].

$$\Psi(Y_i, Y_j) = \psi(Y_i, Y_j) = \exp\left\{-\beta(Y_i - Y_j)^2\right\}$$

where β is a penalty factor for label disagreement between nodes. For inferencing the most likely labels of individual parking spot hypotheses in a given aerial image, we use loopy belief propagation because it is easy to implement [Yedidia *et al.*, 2002].

4 Experimental Results

The goal of this work is to extract the structure of a parking lot that is visible in an aerial image. The knowledge of the image coordinates of parking spots facilitates estimation of parameters that describe the structure of the parking lot. Thus the purpose of our experiments is to verify how well our methods perform in detecting all of the visible parking spots in an aerial image.

	fn	*fp*	*acc*
Intial Estimates	0.6373	0.0298	0.6755
Generated Hypotheses	0.1123	0.3812	0.7399

Table 1: Accuracy comparison of parking spot hypotheses generated by the low-level and high-level analysis layers is measured by three different performance metrics. These metrics include "false negative (*fn*)," "false positive (*fp*)," and "accuracy (*acc*)." Each of these metrics is defined in terms of the entries in a contingency table tabulating the four possible outcomes of a binary classification result.

We use thirteen aerial images collected from the *Google* [3] map service. There are on average 147 visible parking spots in each individual image and a total of 1,912 parking spots across all aerial images.

Table 1 shows the micro-averaged accuracy of the initial proposal and the hypothesis generation. This micro-averaged performance is computed by merging contingency tables across the thirteen different images and then using the merged table to compute performance measures. Since the initial proposal has a false positive rate of nearly zero (2.98%), its parking spot estimates are used as positive examples for training all filtering methods. An equal number of negative examples are randomly generated.

A false positive is a non-parking-spot example that is classified as a parking spot. A false positive output is quite risky for autonomous robot driving; in the worst case, a false positive output might make a robotic vehicle drive somewhere that the robot should not drive. Despite having nearly zero false positives, the canonical parking spots detected by the low-level analysis cover only 37.65% of the true parking spots (720 out of 1,912 true parking spots.) This high false negative rate may cause additional problems for autonomous driving: an autonomous robotic vehicle won't be able to park itself even if there are plenty of parking spots available. By using information provided by the low-level analysis, the high-level hypothesis generation analysis reduces the false negative rate from 63.73% to 11.23%. However, it increases the false positive rate to 38.12% as well. The filtering stage then corrects this shift in false positive rate by removing erroneous hypotheses. Importantly, as we will see in the results, this technique cannot recover from false negatives[4] in the hypothesis generation. However, the false negative rate in the hypothesis generation phase of the high-level analysis is generally low and does not significantly detract from the accuracy.

Table 2 compares the performance of self-trained filtering methods. The parking spot hypotheses generated by the high-level layer were labeled by hand for testing. Hyper-parameters of SVMs were determined by 10-fold cross validation.[5] Eigenspots are computed using positive examples. For the MRF inference, we build a mesh from the estimated layout of parking spot hypotheses where a node in the grid corresponds to an image patch. We again use positive and negative examples to obtain a GMM and use the obtained GMM to estimate node potentials. We observe the results by varying β in the range 0 to 10 with steps of size 2 and the best result was achieved when β is 2.[6]

In table 2, there are three different experimental scenarios. In the first scenario, we trained the filtering methods using a self-supervised set of examples from the image under analysis consisting of the self-labeled positive examples and randomly generated negative examples. In the second scenario, we trained these methods using self-supervised examples from all other images not including the target image. Finally, in the last scenario we trained the methods using self-supervised examples from all images. The randomly generated negative examples were sampled while running each of these scenarios. Due to this randomness in negative examples, we averaged our results over 5 separate runs for each scenario. Each cell in the table displays the mean and standard deviation.

We additional manually generated 1,079 parking spot patches across all thirteen images (averaging 83 parking spots per image). We re-ran the above experiments using these human-extracted parking spot patches. The numbers in parentheses indicate the performance difference between self-supervised and supervised parking spot hypothesis filtering tasks. Positive values in the accuracy column indicate improvements of self-supervised learning over supervised learning whereas negative values in false positive and negative columns indicate improvements. Surprisingly, the algorithm performed slightly worse at times when trained using the more accurately generated manual examples. This likely occurs because the distribution of the test parking spots is created by our hypothesis generation approach.

Ideally, a method would achieve both the lowest false positive and negative rates, but in practice it is hard to achieve both simultaneously. For our autonomous driving application, we prefer the method with the lowest false positive to one with lowest false negative because a false positive is more risky than a false negative. Our results show that our MRFs approach demonstrates superior performance across most experiments. As discussed in Section 3.2, MRFs utilize higher-level interactions to improve prediction accuracy. However, estimating the GMM requires a substantial amount of data; the performance degradation in the first row of the table indicates that the canonical parking spots extracted by the low-level analysis alone were too few to accurately fit this model. Additionally, training an SVM using the subspace generated by the Eigenspots analysis performs only marginally better than simply using the Eigenspot distance measure computation. This performance difference can potentially be decreased by statistically fitting the threshold value used during distance measurement classification. Finally, linear SVMs with no additional preprocessing performed surprisingly well, particularly in terms of false positives.

5 Conclusions

This work proposes a two layer hierarchical algorithm for analyzing the structure of parking lots seen in overhead satellite images. The low-level analysis layer extracts a set of easily detected canonical parking spots and estimates parking blocks using line detection and clustering techniques. The high-level analysis then extends those spots using geometrical characteristics of typical parking lot structures to interpolate and extrapolate new hypotheses and uses self-supervised machine learning techniques to filter out false positives in the proposed hypotheses. Our experiments show that training the classifiers using the self-supervised set of canonical parking

[3]http://map.google.com

[4]A false negative is a parking-spot example that is classified as a non-parking-spot example.

[5]For SVM implementation, we use libsvm which is publicily available at http://www.csie.ntu.edu.tw/∼cjlin/libsvm/

[6]We fit our Gaussian Mixture model using the publicly available GMMBayes from http://www.it.lut.fi/project/gmmbayes/

	Methods	*false negative*	*false positive*	*accuracy*
1	SVMs	0.4425 ± 0.0089 (-0.1214)	0.1525 ± 0.0184 (+0.0332)	0.6592 ± 0.0035 (+0.0589)
	Eigenspots	0.3169 (-0.0811)	0.5420 (+0.1711)	0.5891 (-0.0257)
	SVMs w/ Eigenspots	0.4055 ± 0.0142 (-0.0137)	0.2097 ± 0.0106 (+0.0471)	0.6644 ± 0.0123 (+0.0048)
	MRFs w/ GMM	0.4356 ± 0.0162 (-0.2233)	0.2020 ± 0.0328 (+0.0523)	0.6482 ± 0.0187 (+0.0933)
2	SVMs	0.5093 ± 0.0160 (-0.0208)	**0.0098 ± 0.0008** (+0.0011)	0.8997 ± 0.0026 (-0.0174)
	Eigenspots	0.1831 (-0.0134)	0.1127 (+0.0283)	0.8745 (-0.0251)
	SVMs w/ Eigenspots	0.5094 ± 0.0184 (-0.0125)	0.0109 ± 0.0011 (-0.0016)	0.8988 ± 0.0029 (-0.0162)
	MRFs w/ GMM	0.4048 ± 0.0196 (-0.1752)	0.0167 ± 0.0014 (+0.0041)	0.9130 ± 0.0045 (+0.0064)
3	SVMs	0.4225 ± 0.0081 (-0.0603)	0.0136 ± 0.0005 (+0.0044)	0.9170 ± 0.0016 (-0.0107)
	Eigenspots	**0.1789** (-0.0158)	0.1066 (+0.0285)	0.8811 (-0.0253)
	SVM w/ Eigenspots	0.4135 ± 0.0053 (-0.0344)	0.0159 ± 0.0013 (+0.0022)	0.9167 ± 0.0010 (-0.0119)
	MRFs w/ GMM	0.3539 ± 0.0108 (-0.1831)	0.0212 ± 0.0022 (+0.0076)	**0.9224 ± 0.0023** (+0.0056)

Table 2: Results comparing different filtering methods.

spots successfully adapts the filter stage to the particular characteristics of the image under analysis.

Our experiments additionally demonstrate that additional *prior* parking spot data collected across multiple additional overhead parking lot images offer the learner useful information resulting in increased performance. These examples provide the learner with important demonstrations of occlusions and illumination variations not found in the canonical parking spots extracted by the low-level analysis. Unfortunately, simply adding the examples to the training set is a limited technique for utilizing prior information since training time grows with the number of examples. Thus, this way of using prior information is arguably sub-optimal. Future work will compare this data augmentation technique to alternative techniques from the machine learning literature for incorporating prior information, such as regularizing around prior weight vectors and Bayesian linear regression. Since a high false negative rate might lead our system to underestimate the actual area of a parking lot, we will working on increasing the coverage of our algorithms. We additionally plan to experiment with more sophisticated feature extraction techniques, including well-studied image feature representations such as Histograms of Oriented Gradients (HOG).

6 Acknowledgements

This work is funded by the GM-Carnegie Mellon Autonomous Driving Collaborative Research Laboratory (CRL).

References

[Bishop, 2006] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[Fabian, 2008] Tomas Fabian. An algorithm for parking lot occupation detection. In *Proceedings of IEEE Computer Information Systems and Industrial Management Applications*, pages 165–170, 2008.

[Huang *et al.*, 2008] Ching-Chun Huang, Sheng-Jyh Wang, Yao-Jen Chang, and Tsuhan Chen. A bayesian hierarchical detection framework for parking space detection. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 2097–2100, 2008.

[Kahn *et al.*, 1990] P. Kahn, L. Kitchen, and E.M. Riseman. A fast line finder for vision-guided robot navigation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(11):1098–1102, 1990.

[Li, 2000] Stan Z. Li. *Markov Random Fields Modeling in Computer Vision*. Springer-Verlag, 2000.

[Lieb *et al.*, 2005] David Lieb, Andrew Lookingbill, and Sebastian Thrun. Adaptive road following using self-supervised learning and reverse optical flow. In *Proceedings of Robotics Science and Systems*, 2005.

[Sofman *et al.*, 2006] Boris Sofman, Ellie Lin, J. Andrew Bagnell, Nicolas Vandapel, and Anthony Stentz. Improving robot navigation through self-supervised online learing. In *Proceedings of Robotics Science and Systems*, 2006.

[Stavens and Thrun, 2006] David Stavens and Sebastian Thrun. A self-supervised terrain roughness estimator for off-road autonomous driving. In *Proceedings of Conference in Uncertainty in Artificial Intelligence*, 2006.

[Turk and Pentland, 1991] Matthew Turk and Alex Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

[Urmson *et al.*, 2008] Chris Urmson, Joshua Anhalt, and et al. Autonomous driving in urban environments: Boss and the urban challenge. *Journal of Field Robotics: Special Issues on the 2007 DARPA Urban Challenge*, pages 425–466, 2008.

[Wang and Hanson, 1998] Xiaoguang Wang and Allen R. Hanson. Parking lot analysis and visualization from aerial images. In *Proceedings of the IEEE Workshop on Applications of Computer Vision*, pages 36–41, 1998.

[Wu *et al.*, 2007] Qi Wu, Chingchun Huang, Shih yu Wang, Wei chen Chiu, and Tsuhan Chen. Robust parking space detection considering inter-space correlation. In *Proceedings of IEEE International Conference on Multimedia and Expo*, pages 659–662, 2007.

[Yedidia *et al.*, 2002] Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. Understanding belief propagation and its generalizations. Technical Report TR2001-022, Mitsubishi Electric Research Laboratories, 2002.