The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

Utilizing Prior Information to Enhance Self-Supervised Aerial Image Analysis for Extracting Parking Lot Structures

Young-Woo Seo
Robotics Institute
Carnegie Mellon University
5000 Forbes Ave, Pittsburgh PA 15213, USA
young-woo.seo@ri.cmu.edu

Chris Urmson
Robotics Institute
Carnegie Mellon University
5000 Forbes Ave, Pittsburgh PA 15213, USA
curmson@ri.cmu.edu

***Abstract*— Road network information (RNI) simplifies autonomous driving by providing strong priors about driving environments. Its usefulness has been demonstrated in the DARPA Urban Challenge. However, the need to manually generate RNI prevents us from fully exploiting its benefits. We envision an aerial image analysis system that automatically generates RNI for a route between two urban locations. As a step toward this goal, we present an algorithm that extracts the structure of a parking lot visible in an aerial image. We formulate this task as a problem of parking spot detection because extracting parking lot structures is closely related to detecting all of the parking spots. To minimize human intervention in use of aerial imagery, we devise a self-supervised learning algorithm that automatically obtains a set of canonical 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. To remedy this insufficient positive data problem, we utilize self-supervised parking spots obtained from other aerial images as prior information and a regularization technique to avoid an overfitting solution.**

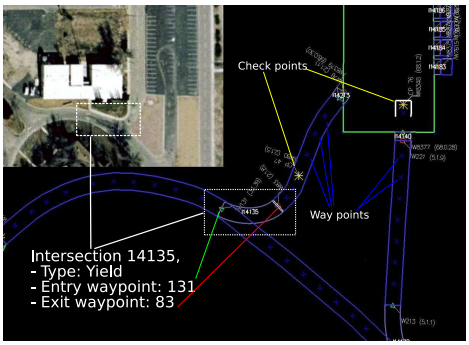


Fig. 1. An example of the road network information used in the 2007 DARPA Urban Challenge. RNI represents an urban environment as a graph that is comprised of a set of vertices (i.e, waypoints marked by "x" and checkpoints marked by "*") and their connections. An intersection is a region that includes a subset of waypoints between entry and exit points.

I. INTRODUCTION

Road network information (RNI) is an essential component for the recent successful autonomous robotic vehicles [17]. In our case, RNI is a topological feature map of an urban environment that informs an autonomous robotic vehicle where it can drive, models of what can be expected, and contextual cues that influence driving behaviors. Figure 1 shows road network information describing the starting chute of the DARPA Urban Challenge site with a corresponding aerial image. This information enables our robotic vehicle to anticipate information about upcoming intersections (e.g., that the first intersection labeled as "I14135" after leaving the staring chute is a yield-intersection) and other fixed rules of the road (e.g., the speed limit is 30 miles per hour).

To navigate, our vehicle chooses a globally optimal route to the goal using the RNI and executes an autonomous driving loop: it initiates behaviors based on available RNI (e.g., *handle_intersection or drive_down_lane*); it analyzes onboard sensors' outputs to interpret its surroundings (e.g., estimating the current pose or perceiving static and dynamic obstacles on drivable regions); it executes motions to achieve the goal of the current behaviors (e.g., arriving at a particular waypoint by driving on a road lane). In this loop of actions, RNI simplifies autonomous driving in that it allows a robotic vehicle to focus its attention on drivable regions which require a detailed analysis, neglecting less important regions [17].

Road network information is currently manually generated using a combination of GPS survey and aerial imagery. These techniques for converting digital imagery into road network information are labor intensive and error-prone. To fully exploit the benefits of road network information, these processes should be automated.

We envision an aerial image analysis system that automatically generates RNI for a route. In this paper, as a step toward this goal, we present an aerial image analysis algorithm that extracts the structure of parking lots. Without representing this structure, our motion planner has to superimpose virtual structures onto parking lots and evaluate all the feasible patches over the structures [6]. This two step process (create virtual structure and apply nominal motion planning algorithms) is computationally expensive and can be avoided by generating the structure offline, a priori.

Although parking lot structures in aerial images are readily recognizable, automatically extracting structures is challenging because their shapes are *only approximately regular* and aerial image acquisition process is *noisy*. Specifically, 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. 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 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 III.

II. Related Work

Overhead aerial images have been utilized to provide prior information about environments for outdoor navigation robots. Despite issues with age, aerial image analysis provides an alternative, but important structural overview of operational environments that enables robots to plan globally to achieve their goals. In combination with other onboard sensors such as vision sensors and range finders, aerial images have been used for generating cost maps for long-range traversal [13], [14], global localization [3], building and maintenance of robots' world model [12], [17], and mapping [11].

Overhead imagery has thus been used as a complement to onboard sensor data to guide outdoor robot navigation. To the best of our knowledge, there is no work in automatically generating road network information from overhead aerial images.

There are two similar works in the realm of parking structure analysis. Wang and Hanson propose an algorithm that uses multiple aerial images to extract the structure of a parking lot for simulation and visualization of parking lot activities [18]. Multiple images from different angles are used to build a 2.5 dimensional elevation map of a 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. Dolgov and Thrun present algorithms that build a lane-network graph of a parking lot from sensor readings [4]. They first build a grid map of static obstacles from range measurements of a parking lot and use a Markov Random Field to infer a topological graph that most likely fits the grid map. This work is very close to ours in that they building a road network for their robotic vehicle, but different in that they need to drive the robot to collect range measurements, which are the sole input to their mapping algorithm.

Most prior work in parking spot extraction [5], [7], [19] 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 entire parking lot structure from overhead imagery. A similarity between our work and these works on empty parking spot detection lies in the fact that we utilize coherent structural patterns over entire image region.

The problem of how best to frame self-supervised learning problems has recently attracted 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, a conventional laser range finder provides quite accurate distance estimates between a robot and surrounding objects, but their ranges are limited. Sofman et al use those local range estimations as self-labeled examples to learn relations between the characteristics of local terrain and corresponding regions in aerial images [14]. These learned relations were used to map aerial images to long range estimates of traversability over regions that a robot is going to explore. Similarly, Stavens and Thrun utilize laser range measurements to predict terrain roughness [15]. 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 et al 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 [10]. In our algorithms, the 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.

III. Aerial Images Analysis for Extracting Parking Lot Structures

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. Figure 2 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 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 2). 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 figure 2).

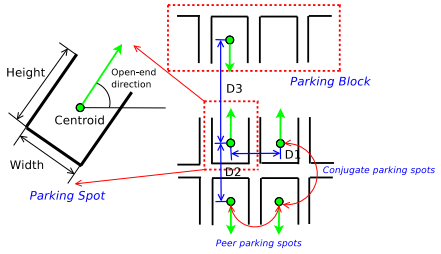
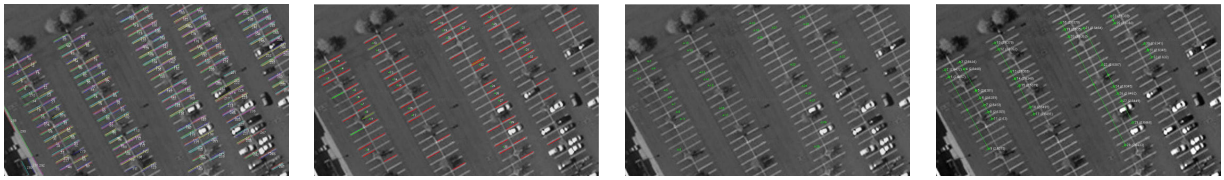


Fig. 2. This illustration depicts the parking spot and parking block representations used throughout this work.

If the image locations of all visible parking spots are known, it would be trivial to estimate the alignment and block parameters shown in the figure 2. 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 III-A presents the multiple 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 III-B 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 in Section III-B.2, which removes erroneous hypotheses from the collection.



(a) Results of a line extraction. (b) Results of a line clustering and filtering. (c) Results of the initial parking spot estimates. (d) Results of parking block discovery.

Fig. 3. From the left, these figures present a sequential order of image processing steps in the low-level image analysis.

A. Low-Level Analysis: Detecting Canonical Parking Spots

Geometrical and image characteristics differ between parking lots. Most overhead aerial 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 [8]. This approach computes image derivatives to obtain intensity gradients at each pixel and quantizes the gradient directions over 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. Figure 3(a) shows an illustrative example image with a set of extracted lines.

Although a majority of extracted lines may align with lane markings of the underlying parking lot, some of them come from other image regions such as road lanes or contours of other adjacent buildings. Since we only need the lines aligned with the line-markings of the parking lot, it is necessary to remove lines that do not represent 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. Figure 3(b) shows a set of candidate lines used for parameter estimation.

For the parameter estimation, we first estimate the nominal height of 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 each parking spots' open-end.

The completion of these image processing steps results in generating few, but highly accurate initial estimations of true parking spots. Figure 3(c) shows the image location of parking spots found using this low-level analysis.

This low-level analysis is then extended to additionally identify entire parking blocks. To achieve this, we project the centroids of all the initial parking spots onto a virtual line whose orientation is the mean of the initial parking spots' orientations. 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 a 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 2) is used to determine boundaries between parking blocks. Figure 3(d) shows seven distinct parking blocks discovered by this analysis. 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 2).

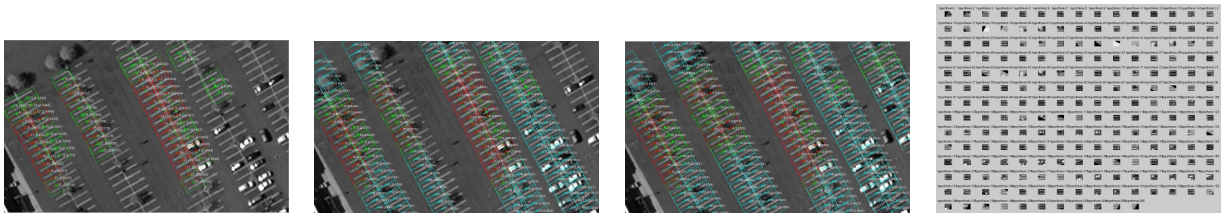
B. High-Level Analysis: Interpolation, Extrapolation, Block Prediction, and Filtering

The high-level layer is intended to detect all the visible parking spots in an image. To this end, it first hypothesizes the 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.

1) Parking Spot Interpolation and Extrapolation: A parking spot hypothesis represents an image location that indicates the centroid of a potential parking spot. A rectangular image patch around the hypothesis is evaluated to determine if a local characteristic of the image is similar to that of a true parking spot. To cover the set of image regions that possibly contain true parking spots, we use the image coordinates of centroids of the initial parking spots as the starting points in each of the discovered parking blocks. We then generate parking spot hypotheses by selecting image locations through three processes: interpolation (Figure 4(a)), extrapolation (Figure 4(b)), and block prediction (Figure 4(c)). The interpolation procedure chooses image regions within a parking block, whereas the extrapolation procedure extends hypotheses beyond estimated parking block boundaries. 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.

2) 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 4(d) shows a complete set of the generated parking spot hypotheses. Each of 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, $\mathbf{g}_i$, where $\mathbf{g}_i$ is an $m = |\text{height} \times \text{width}|$-dimensional column vector. Because raw intensity values of grayscale image patches are inconsistent, even in the same class, we use three different pieces of information to inject invariance into parking spot representation: intensity statistics of a patch such as mean, variance, skewness, smoothness, uniformity, and entropy; responses of Radon transform; local histograms of oriented gradients (HOG) [2].

Our experiments compare four machine learning techniques for this binary classification task: Support Vector Ma-



(a) Results of interpolation. (b) Results of extrapolation. (c) Results of block prediction. (d) There are 206 parking pots hypotheses generated.

Fig. 4. From the left, these figures show a sequence of hypothesis generation procedure.

chines (SVMs), Eigenspots, Markov Random Fields (MRFs), and Bayesian Linear Regression (BLR).

Support Vector Machine. SVMs are the *de facto* supervised learning algorithm for binary classification. They seek to find the hyperplane that is maximizing a notion of margin between each class [1]. Linear SVMs are fast, have publicly available implementations, and handle high-dimensional feature spaces well.

Eigenspots. Since processing these high-dimensional image patches is computationally expensive, we reduce the dimensionality of our vector space by using principal component analysis (PCA) [1] to find the principal subspace of the initial canonical parking spots found by the low-level analysis; we retain the top $k \ll m$ dimensions of the space. In homage to Turk and Pentland [16], 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. Our first technique simply measures the distance from a candidate patch to the center of the space (i.e. the mean canonical parking spot, $\mathbf{\Psi}$). Given a new image patch $\mathbf{g}$, we compute, $T(\mathbf{g}) = \|\mathbf{D}^{-1/2}\mathbf{E}^T(\mathbf{g}-\mathbf{\Psi})\|^2$ where $\mathbf{\Psi} = \frac{1}{|\text{number of positives}|}\sum_i \mathbf{g_i}$, $\mathbf{D}$ is a diagonal matrix containing eigenvalues $\lambda_1, ...\lambda_k$, and $\mathbf{E}$ is a matrix whose columns are the eigenvectors of the covariance matrix used in the PCA computation. $T(\mathbf{g})$ is also known as the Mahalanobis distance [1] 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 usage simply pushes the examples through the PCA transformation before training a 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. Because SVMs and Eigenspots only consider the local characteristics of an image patch to perform the binary classification, their performances are limited by the distribution of the training data. Thus 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) [9]. A pairwise MRF $\mathcal{H}$ is an undirected graphical model that factorizes the underlying joint probability distribution $P(Y, G)$ by a set of pairwise cliques. [1] $\mathcal{H}$ is comprised of a set of nodes and their edges where a node models a random variable and an edge between nodes represents dependence between them.

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 Y (i.e. whether a hypothesis ($\mathbf{g}_i$) is parking spot ($y_i = 1$) or not) given the structure of the undirected graph, $\mathcal{H}$, and characteristics of image patches, G. The joint probability distribution is modeled as

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

where $\Phi(G_i, Y_i)$ is a node potential, $\Psi(Y_i, Y_j)$ is an edge potential, Z is the partition function that ensures a probability density of this model, $N(i)$ is the set of nodes in the neighborhood of the ith node. Our implementation of MRFs consider first-order neighbors.

As we assume that candidate parking spots are generated from a mixture of multivariate Gaussian distributions, we estimate the node potentials using a Gaussian Mixture model (GMM) [1]. Due to the possibility of two class 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 Potts model [9]. 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 [20].

Bayesian Linear Regression Our self-supervised canonical parking spots are highly accurate, but the quantity is often too few to generalize over unseen image patches. To remedy this insufficient number of positive examples, we use canonical parking spots previously obtained from other aerial images. As we will show its benefit in the experimental results, this certainly helps our hypothesis filters improve their performances. However, naively consuming all the available data might result in an overfitted solution. To effectively utilize data, we employ a Bayesian linear regression (BLR). BLR provides a theoretical way of incorporating previously obtained parking spot templates as a prior information for the optimal weight vector learning. The optimal weight vector, $\mathbf{w}^*$, is obtained by

$$p(\mathbf{w}^*|\mathbf{G}) \propto \arg\max_{\mathbf{w}} p(\mathbf{G}|\mathbf{w})p(\mathbf{w})$$

where the likelihood function is modeled by $p(\mathbf{G}|\mathbf{w}) = \prod_{i=1}^{N} p((\mathbf{g}_i, y_i)|\mathbf{w}) \propto \exp\left\{\frac{1}{2\sigma^2}\sum_i (y_i - \mathbf{w}^T\mathbf{g}_i)^2\right\}$ and the prior distribution is given as a zero-mean Gaussian, $p(\mathbf{w}) \propto \exp\left\{-\frac{1}{2}\mathbf{w}\mathbf{\Sigma}^{-1}\mathbf{w} + \mu^T\mathbf{\Sigma}^{-1}\mathbf{w}\right\}$. The final form of BLR is a regularized linear regression where the parameters of the resulting conditional Gaussian distribution of $\mathbf{w}^*$ given data D is

$$\begin{aligned}\mathbf{\Sigma}_{\mathbf{w}|\mathbf{D}} &= \left(\mathbf{G}\mathbf{G}^T + \lambda\mathbf{I}\right)^{-1} \\ \mu_{\mathbf{w}|\mathbf{D}} &= \left(\mathbf{G}\mathbf{G}^T + (\sigma^2\lambda)\mathbf{I}\right)^{-1}\mathbf{Y}\mathbf{G}\end{aligned}$$

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

where λ is a regularizing term that controls contributions of the weight prior. We classify an image patch $\mathbf{g}_i$ as;

$$h(\mathbf{g}_i) = 2I\left[y(\mathbf{g}_i) \geq \delta\right] - 1, \delta \in \mathbf{R}.$$

where $y(\mathbf{g}_i)$ is the output of BLR and $I\left[y(\mathbf{g}_i) \geq \delta\right]$ is an indicator function that returns 1 if $y(\mathbf{g}_i)$ is greater than δ, otherwise 0.

IV. 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 the visible parking spots in an aerial image.

	fn	*fp*	*acc*
Self-supervised Parking Spots	0.6373	0.0298	0.6755
Generated Hypotheses	0.1123	0.3812	0.7399

TABLE I

Accuracy comparison of parking spot hypotheses generated by the low-level and high-level analysis layers is measured by four different performance metrics. These metrics include "false negative (*fn*)," "false positive (*fp*)," and "accuracy (*acc*)."

We use thirteen aerial images collected from *Google* [2] map service. There are on average 147 visible parking spots in each individual images and a total of 1,912 parking spots across all aerial images.

Table I shows the micro-averaged accuracy of the self-supervised canonical parking spots 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 self-supervised examples has a very low false positive rate (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 self-supervised parking spots obtained 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 [3] 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 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 II 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.[4] 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 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.[5] For BLR, we found that the best results is obtained when λ is 5 and use .5 as a threshold for binary classification.

In the table II, there are three blocks of rows describing 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.

In addition, we 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 indicates the performance difference between self-supervised and supervised parking spot hypothesis filtering tasks. Positive values in the accuracy 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 the test distribution is that created by our hypothesis generation approach.

Ideally, the method with the lowest false positive and negative rates would be the best, but in practice it is hard to achieve both of them simultaneously. For our autonomous driving application, we prefer the method with the lowest false positive to the one with lowest false negative because a false positive is more risky than a false negative. In general, the performances of hypothesis filters are improved as the number of training data is increased. Linear SVMs performed surprisingly well, particularly in terms of false positives and accuracy. Additionally, training an SVM using the subspace generated by the Eigenspots analysis performs only marginally better than simply using the Eigenspot distant measure computation. This performance difference can potentially be decreased by statistically fitting the threshold value used during distance measure classification. As discussed in Section III-B.2, MRFs utilize higher-level interactions to improve prediction accuracy. However, estimating

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

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

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

[5] We fit our Gaussian Mixture model using the publicly available GMM-Bayes from http://www.it.lut.fi/project/gmmbayes/

		false negative	*false positive*	*accuracy*
Only self-supervised examples	SVMs	0.4370 ± 0.0141 (-0.2385)	0.1821 ± 0.0104 (0.0397)	0.6523 ± 0.0115 (0.1436)
	Eigenspots	0.2861 (0.0145)	0.5745 (-0.1240)	0.6019 (0.0271)
	SVMs w/ Eigenspots	0.4382 ± 0.0102 (-0.2200)	0.2385 ± 0.0296 (0.0785)	0.6313 ± 0.0093 (0.0930)
	MRFs w/ GMM	0.4493 ± 0.0162 (-0.2533)	0.2366 ± 0.0328 (0.0490)	0.6298 ± 0.0247 (0.1420)
	BLR	0.1882 ± 0.0060 (-0.2715)	0.5861 ± 0.0169 (0.1756)	0.6242 ± 0.0073 (0.0632)
Only prior	SVMs	0.5364 ± 0.0050 (-0.1894)	**0.0111** ± **0.0000** (0.0018)	0.8936 ± 0.0016 (0.0005)
	Eigenspots	0.1763 (-0.0122)	0.1197 (0.0429)	0.8701 (-0.0372)
	SVMs w/ Eigenspots	0.6324 ± 0.0094 (-0.1730)	0.0158 ± 0.0001 (0.0044)	0.8724 ± 0.0001 (-0.0031)
	MRFs w/ GMM	0.6112 ± 0.0206 (-0.2071)	0.0187 ± 0.0024 (0.0090)	0.8744 ± 0.0002 (-0.0007)
	BLR	0.2631 ± 0.0216 (-0.2668)	0.0728 ± 0.0029 (0.0439)	0.8927 ± 0.0022 (-0.0070)
Self-supervised + prior	SVMs	0.4130 ± 0.0041 (-0.2522)	0.0176 ± 0.0000 (0.0080)	**0.9153** ± **0.0016** (0.0121)
	Eigenspots	**0.1757** (-0.0112)	0.1115 (0.0375)	0.8776 (-0.0334)
	SVM w/ Eigenspots	0.5278 ± 0.00081 (-0.2360)	0.0208 ± 0.0013 (0.0083)	0.8931 ± 0.0010 (0.0051)
	MRFs w/ GMM	0.4974 ± 0.0098 (-0.2888)	0.0216 ± 0.0012 (0.0111)	0.8976 ± 0.0003 (0.0113)
	BLR	0.2014 ± 0.0039 (-0.3021)	0.0892 ± 0.0083 (0.0618)	0.8918 ± 0.0063 (-0.0175)

TABLE II

RESULTS COMPARING DIFFERENT FILTERING METHODS.

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 this fit this model. Finally, BLR demonstrates its strength of utilizing prior information when more data is used for training; it shows a lower rates across all three performance metrics.

V. CONCLUSIONS

This work proposes a two layer hierarchical algorithm for analyzing the structure of parking lots visible in overhead aerial 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 state-of-the-art 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 spots extracted by the low-level analysis successfully adapts the filter stage to the particular characteristics of the image under analysis.

Our experiments additionally demonstrate that additional data from 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. However, simply adding more examples to the training set results in a data-overfitted solution. By employing Bayesian linear regression, we show an effective technique to utilize the prior information.

In future work we will consider parking lots with more complex geometries.

VI. ACKNOWLEDGMENTS

This work is funded by the GM-Carnegie Mellon Autonomous Driving Collaborative Research Laboratory (CRL). The authors thank Nathan Ratliff for many helpful and insightful comments.

REFERENCES

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

[2] Navneet Dalal and Bill Triggs, Histograms of oriented gradients for human detection, In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 886-893, 2005.

[3] C.U. Dogruer, B. Koku, and M. Dolen, Global urban localization of outdoor mobile robots using satellite images, In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3927-3932, 2008.

[4] Dmitri Dolgov and Sebastian Thrun, Autonomous driving in semi-structured environments: Mapping and planning, In *Proceedings of IEEE International Conference on Robotics and Automation*, pp. 3407-3414, 2009.

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

[6] David Ferguson, Tom M. Howard, and Maxim Likhachev, Motion planning in urban environments: Part II, In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1070-1076, 2008.

[7] Ching-Chun Huang and Sheng-Jyh Wang and 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*, pp. 2097-2100, 2008.

[8] 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*, Vol. 12, No. 11, pp. 1098-1102, 1990.

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

[10] 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.

[11] Martin Persson, Tom Duckett, and Achim Lilienthal, Improved mapping and image segmentation by using semantic information to link aerial images and ground-level information, *Recent Progress in Robotics*, pp. 157-169, 2008.

[12] Chris Scrapper, Ayako Takeuchi, Tommy Chang, Tsai Hong, and Michael Shneier, Using a priori data for prediction and object recognition in an autonomous mobile vehicle, In *Proceedings of the SPIE Aerosense Conference*, 2003.

[13] David Silver, Boris Sofman, Nicolas Vandapel, J. Andrew Bagnell, and Anthony Stentz, Experimental analysis of overhead data processing to support long range navigation, In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2443-2450, 2006.

[14] 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.

[15] 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.

[16] Matthew Turk and Alex Pentland, Eigenfaces for recognition, *Journal of Cognitive Neuroscience*, Vol. 3, No. 1, pp. 71-86, 1991.

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

[18] 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*, pp. 36-41, 1998.

[19] 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*, pp. 659-662, 2007.

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