A Hierarchical Image Analysis for Extracting Parking Lot Structures from Aerial Images

Young-Woo Seo and Chris Urmson

CMU-RI-TR-09-03

January 2009

Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

Abstract

The availability of road network information simplifies autonomous driving by providing useful prior information about driving environments which is valuable for planning and perception. It tells a robotic vehicle where it can drive, models of what can be expected where, and provides contextual cues that influence driving behaviors. Currently, however, road network information for driving environments 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. As a step toward this goal, we present an algorithm that extracts the structure of a parking lot visible from a given aerial image. We propose a hierarchical approach to generating and evaluating candidate hypotheses. We test three different machine learning algorithms and their combinations for removing erroneous hypotheses. From the experimental results, our Markov Random Field implementation performs best in terms of false negative rate and Eigenspots performs best in terms of false positive rate.

Contents

1 Introduction

The 2007 DARPA Urban Challenge demonstrated the potential for driverless automobiles to operate in urban daily life in the near future [28]. Important to the success of the Urban Challenge was the availability of a detailed digital road network (or map). The road network information tells a robotic vehicle where it can drive and provides contextual cues that influence the driving behavior. For example, the road network information lets the robotic vehicle know information about upcoming intersections (e.g. that the intersection is a four-way stop and that the robot must conform to precedence rules) and other fixed rules of the road (e.g. speed limits). Currently the road network information about the driving environment is manually generated using a combination of GPS survey and aerial imagery. If the robot localizes itself with respect to the same coordinate system, it knows how it can navigate the environment by first relating its current location to the map and then driving relative to roads described in the map. Figure 1 shows an illustrative example where a robotic vehicle is entering a parking lot in a mall and is looking for an empty parking spot. According to the road network information (arrowed line and stars), the robot generates and follows a trajectory (blue dashed line) while perceiving moving objects in real-time. Since the robot knows where it can park, based on the road network information, what the robot needs to do next is to find one of the empty parking spots by using its perception system.

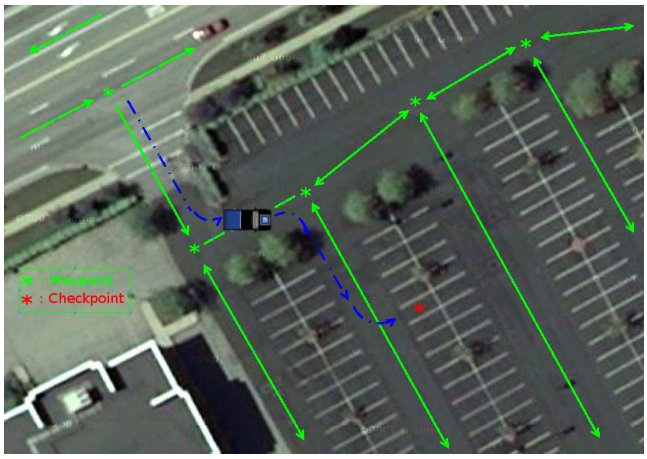


Figure 1: An illustrative example shows how the information of road network is used in autonomous driving. A rectangle with blue and black represents an autonomous robotic vehicle which follows a blue trajectory based on the information of the road network depicted in green straight lines. The green stars are waypoints that connect edges in the road network and the red star is a checkpoint that the robot must visit for accomplishing its task.

Techniques for converting digital images into road network information are labor-intensive and hence error-prone. Therefore it would be very useful to automatically

generate the road network information from aerial images. In this paper, as a step toward building such a property-map of road network, we present a hierarchical approach for extracting the structure of a parking lot from a given aerial image.

When looking at aerial images of parking lots, the structure is readily recognizable. Hence the goal of extracting parking lot structures might be considered easily achievable. However it is challenging due to primarily the fact that image acquisition process is inherently noisy. In this work, parking spot is the smallest unit of a parking lot. Detecting all the visible parking spots in a given image extracts much of the parking lot structure. Thus one might consider template matching as an appropriate approach because the shape of parking spots in the real world is consistent to one another at least in the same parking lot. In a typical setting of the template matching [5], we need a set of canonical templates that is used to compare to any candidate image patches. However, it is not easy to obtain a set of good templates of parking spots from a given image because their appearances are not consistent. In other words, their appearances are inconsistent to one another in the given image because of vehicle occupancy, occlusions by other structures such as trees and adjacent buildings, or different illuminations (e.g. under the shade of buildings.) In other words, although the actual width and height of a parking spot in an image might be inferred from the available prior knowledge, their intensity and chromacity values are different due to other objects in the image or different luminance conditions. Another way of handling this detection task is to utilize all the relevant geometric primitives such as straight lines and build up useful structures from the obtained primitives. Structure extraction only fails because:

- Without prior information about boundaries of a particular parking lot, it is hard to distinguish which of the extracted lines are matched to lane-markings of the parking lot;
- If the quality of lane-marking is poor, image-gradient based line extraction algorithm do not work;
- Some of the lines matched to line-markings of parking lots might be occluded by other structures in the image such as trees;
- Finally, the shapes of parking lots are only approximately regular. Their shapes look similar, but they are slightly different. Thus, the structure of parking lots needs to be analyzed individually.

In order to handle these problems effectively, we approach the task of extracting parking lot structures from aerial images hierarchically from low-level image processing procedures through high-level structure inference. The low-level layer performs image processing algorithms to obtain a set of parameters, which captures the geometric characteristics of the underlying parking lot. Using the estimated parameters, the high-level layer infers the structure by first generating parking spot hypotheses and then evaluating rectangular image patches around hypotheses with respect to their local image characteristics and global spatial patternn.

This paper is organized as follows. In section 2 we survey the literature and compare ours to the related work. Section 3 details our hierarchical image analysis for

extracting the parking lot structure from a given aerial image. This includes procedural descriptions of two different layers: low-level aerial image analysis and high-level structure inference. Section describes how we evaluate our method with the collected aerial images. Section 5 summarizes experimental findings and proposes the future work based on what we find from the experiments.

2 Related Work

There is little prior work on parking lot structure analysis from imagery. Wang and Hanson's work is the most similar to ours in that their method analyzes parking lot structure from aerial image [31]. Our approach is different from theirs in that we only use a single aerial image to extract parking lot structure whereas they require multiple aerial images. Furthermore, their goal of parking lot analysis is to simulate and visualize parking lot activities. To detect parking lot structure, they use two different maps: a 2-dimensional intensity map and a 2.5-D elevation map. The two-dimension intensity map provides the layout of individual parking spots whereas 2.5-D elevation maps provides bumpy regions around vehicle locations. By combining these two maps they are able to identify the structure of a parking lot. A drawback of their approach is that their algorithm requires multiple aerial images which have good feature correspondences to generate the elevation map. Although there are high-quality aerial images publicly available, it is not easy to obtain multiple images with different angles on the same geographic location. Their approach is similar to ours in that they utilize periodic spatial patterns to identify parking spots.

There are two similar, but not directly related works that analyze a series of images from a fixed angle surveilance camera to detect empty parking spots. Wu and his colleagues utilize an inter-space dependency between parking spots when differentiating empty parking spots from occupied ones [35]. It would fail to investigate the local characteristics of a parking spot image patch for detecting empty parking spots because of shadow and inter-vehicular occlusions. To cope with these problems, they consider three neighboring parking spots together while determining their occupancy. To determine the availability of parking spots, they train a Support Vector Machines (SVMs) to classify a given three-parking spots patch into one of the eight classes. They define a penalty matrix and utilize a Markov Random Fields (MRFs) to resolve label conflicts between two neighboring patches. Huang and his colleagues tackle the same problem [35] of empty parking spot detection with a different approach. For detecting empty parking spots, they propose a three-layer hierarchical Bayesian network to exploit interdependency between parking spots [9]. To this end, they use the concept of parking blocks in which a parking spot is modeled as an equal-sized cube. Similarily we also discover boundaries of parking blocks from a parking lot image for a better understanding of the parking lot structure. In their Bayesian hierarchical detection framework, the top-layer analyzes individual parking spots based on a local characteristics, the middle-layer is the result of the top-layer's classification with neighboring constraint, and the bottom layer is a binary classification of a parking spot's availability based on higher layers' outputs. Since their camera pose is fixed, it is straightforward to collect training examples for training classifiers based on conditional probabilities of three classes: "vehicle," "ground," and "otherwise." For training a classifier for "vehicle" class, they represent car training data by a combination of a RGB-variant color space and chromatic information in order to reduce variances in luminance. These simple classifiers are used as the top-layer components. For the inference of the bottom layer, they generates a number of pseudo data based on parameters of parking spots' dimension and then learn probability distributions for "empty" and "occupied" classes. By using these pseudo data, they also obtain accumulated histograms for each neighboring pair and

use these histograms to impose constraints between neighboring pairs. Their works are similar to ours in that exploit structural characteristics of parking lots by using graphical models. In other words, they assume that the availability of a parking spot can be inferred from availabilities of neighboring spots. Their works are different from ours in that they know the structure of the parking lot and they have more image data for analysis of local image features.

Our low-level image analysis classifies lane-markings from backgrounds. Lane marking detection is one of the major research areas in the development of autonomous vehicles and mobile robots. The detected lane-markings may be used for driver-assistance system and urban structure analysis from top-view aerial images. A major problem of detecting lane-marking from images is that appearance of lane-markings are not consistent because of occlusions by other objects or illuminational difference. For handling inconsistent appearance in lane-markings, researchers have used different color spaces instead of directly using RGB values. Sun and his colleagues use HSI (Hue-Saturation-Intensity) color space and devised a heuristic to determine when saturation values of images should be used [24]. Li and his colleagues devise a heuristic that converts RGB values into ones, which make lane-markings more sailent [16]. A clustering algorithm is used to identify potential lane-marking image regions from the transformed image and a connected-component algorithm is used to detect lane-markings. Lipski and his colleagues also convert multiple road images in HSI color space into a top-view image and analyze local histograms of color distributions [18]. They combine these color distributions with inputs from other sensors such as lidar and radar to identify lane-markings. Another way of handling this inconsistent appearance problem is to use geometric primitives such as stright lines or curves. Hough transform and its variants have been used to extract straight lines to detect lane-markings [21], [30], [37]. Spline function and its variants have also been used to detect lane-markings and their shapes [32], [33]. We develop a binary classification for lane-marking detection and try to use parking lot structure analysis. Particularly we learn two multivariate Gaussian distributions for "lane-marking" and "non-lane marking" classes and use them to estimate conditional probabilities of a pixel given classes. Wang and Hanson manually extract parking spot lane-markings and use them to identify parking lot lane-markings [31].

For the objection recognition from given images, there have been two major concerns: efficient methods of image features representation and effective recognition methods. The Support Vector Machines has been extensively used as one of the *de facto* standard methods in that it is very flexible to use high-dimensional feature vectors and provide a theoretical sound background. It seeks for a hyperplane that optimally separates example space by maximizing margins between different classes [4]. However since it examines only local characteristics of input data to determine the class of data, its performance is always limited by the quality and the number of data. Turk and Pentland utilize a principal component analysis to effectively represent high-dimensional human face images into a low-dimension without much loss of the original data [27]. They choose k most significant eigenvalues of the covariance matrix of face images and use the corresponding eigenvectors to form a new and low-dimensional face space. They showed a significant reduction of feature dimensions while maintaining the same performance as the one with the full feature space for their face detection task.

Although we formulate the extraction of parking lot structure as parking spot detection, it would very useful to consider the relation between parking spots instead of only investigating them individually. Markov networks and their variants have been extensively studied in computer vision community under the topic of modling such spatial relationships. These techniques include Markov Random Fields [6], [17], [23] or its variants Conditional Random Fields [1], [34] and Discriminitive Random Fields [15]. The fundamental idea of Markov networks is that the value of an image pixel cannot be independent to those of its neighbors. To support this idea, they offers a numberous ways of massaging the joint probability distribution of random variables. In particular, these techniques are flexible in modeling real words problems because they offer the compatibility functions, which are used to model interactions among variables of interest, and provides well-defined techniques for solving learning and inference problems in the undirected graphical models. These techniques have been mainly applied to computer vision and their application areas include augmentation of range measurements by intensity image [3], generation of the optimal navigable path [20], 3D reconstruction of environments from a single image [22], image segmentation [6], object recognition [29], [7], [34], and etc. Although their application areas are different from one to another, the most frequent problem to solve is an inference problem that is to find the most probable state of the world given an image – the most probable explanation (MPE) given data [12]. We utilize a MRF to find the most probable layout of parking spots given an aerial image. Specifically, we hypothesize the locations of the true parking spots by choosing a number of image locations. In our MRF implementation, each of these hypotheses is modeled as a random variable and their joint probability distribution is factorized by undirected graph. We choose an undirected graph to estimate the joint density because it is hard to identify causal directionality among random variables. Since it is also difficult to learn the optimal structure of an undirected graph primarily due to absence of directionality among the variables, we assume that the structure of the undirected graph is given. The given structure allows only pairwise interactions among the variables. Given a particular structure and values of random variables, one of the most interesting problems to solve is to know the most likely state of the world (i.e. Most Probable Explanation). For solving this inference problem, there are three different types of inference techniques: exact inference, sampling-based approximate inference, and variational/belief propagation approximate inference [10], [12]. We choose the loopy belief propagation [6], [36] for solving the most likely labeling on parking spot hypotheses for its simple implementation.

3 Hierarchical Image Analysis for Extracting Structure of Parking Lot

This section details our method for extracting the structure of a parking lot from an aerial image. A structure of a parking lot in an aerial image is characterized by a set of parking blocks and their parking spots. Notice that we are not try to understand the complete structure of a parking lot, but instead identify the structure that is partially visible in a given image. Thus we aim to detect all the visible parking spots as the goal of extracting the structure of a parking lot. Figure 2 illustrates how a parking lot is represented in this study. A parking spot is a subregion of an aerial image in which is used for a single vehicle parking in real world. In our work, it is parameterized by its height, width, orientation of the open-end, and image coordinates of its centroid. A parking block is a row of parking spots of which open-ends are aligned to the same orientation. A parking block is also characterized by a set of parameters imposed by its parking spots and the distance between peer parking spots in the block (i.e. "$D1$" in the 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 the figure 2).

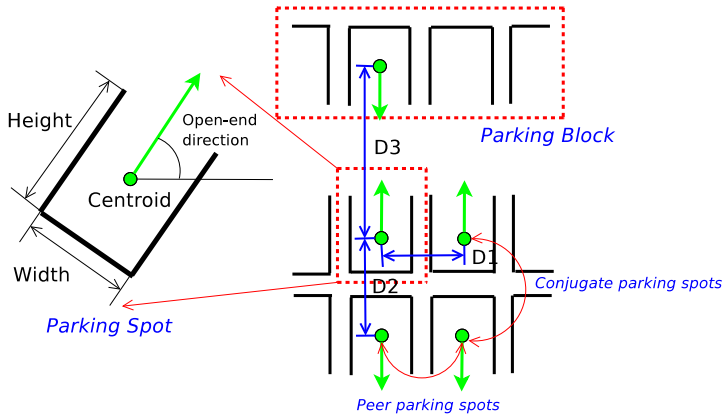


Figure 2: This illustration shows the representations of a parking spot and a parking block in this work.

Ideally estimation of these parameters should provide us with a clear explanation of the structure of the underlying parking lot. However, in practice, the estimated parameters about parking spots would not perfectly explain all the shapes of parking spots in the given image because the appearances of parking spots are inconsistent and the shapes of parking lots are only approximately regular. To tackle these problems effectively, we approach this problem hierarchically from low-level image processing procedures to high-level structure inference. Particularly, the low-level aerial image analysis aims to estimate parameters of a parking lot. The high-level structure inference utilize these parameters to generate parking spot hypotheses and evaluate rectangular

image patches around hypotheses to determine if they are in fact parking spots.

3.1 Low-Level Image Analysis

The low-level image analysis aims to estimate parameters of a parking lot and discover a partial structure of the underlying parking lot. To this end, the low-level aerial image analysis performs low-level image processing procedures: line extraction, line clustering, block prediction, and parameter estimation.

3.1.1 Line Extraction

Geometric primitives such as line or curve are important for structure analysis in images. Straight lines are used in many areas in image analysis and computer vision such as estimating vanishing points [13], relative depths and heights of objects, [8], driving directions [11], relative orientations in indoor [14] or outdoor [19] scene interpretations, etc. We also utilize straight lines to estimate parameters of a parking lot. For extracting lines, we use the approach proposed by [11]. The first step of this method is to compute image derivatives. The intensity gradient at each pixel is used to obtain the magnitude and the direction of the pixel. The k predefined ranges are used to quantize the gradient directions. Pixels whose gradient magnitudes are less than a threshold are filtered out in order to reduce the amount of image computation. A connected component algorithm is then utilized to group the pixels assigned the same direction, in order to form line supporting regions. Specifically, a line supporting region, S_j is a set of x-y coordinates of image pixels that have the same direction of image intensity gradient. A line is parameterized by its orientation, θ, and distance from the origin, ρ. These line parameters are computed from the eigenvalues and eigenvectors of the matrix, D_j, associated with a line supporting region, S_j. That is,

$$\begin{aligned} S_j &= \{(x_{j,1}, y_{j,1}), ..., (x_{j,n}, y_{j,n})\} \\ D_j &= \begin{bmatrix} \sum_r \tilde{x}_r^2 & \sum_r \tilde{x}_r \tilde{y}_r \\ \sum_r \tilde{x}_r \tilde{y}_r & \sum_r \tilde{y}_r^2 \end{bmatrix} \end{aligned}$$

where $\tilde{x} = x_r - \bar{x}_j$ and $\tilde{y} = y_r - \bar{y}_r$ are the dispersion from the mean pixel coordinates, and $\bar{x} = \frac{1}{n}\sum_r x_r$, and $\bar{y} = \frac{1}{n}\sum_r y_r$. Let us denote $\{\lambda_k, \mathbf{v}_k\}_{k=1,...,m}$ the result of eigen analysis of the matrix, D_i, where $\mathbf{v}_k$ is the kth eigenvector and λ_k is its corresponding eigenvalue, respectively. [1] The direction of a line, θ is determined by the eigenvector associated with the largest eigenvalues, $\mathbf{v}_1$.

$$\begin{aligned} \theta &= tan^{-1}(\mathbf{v}_{1,2}, \mathbf{v}_{1,1}), \\ \rho &= \bar{x}\cos\theta + \bar{y}\sin\theta \end{aligned}$$

where $\mathbf{v}_{1,2}$ is the second component of the first eigenvector, $\mathbf{v}_{1,1}$ is the first component of the first eigenvector, and $(\bar{x}, \bar{y})$ is the mid point x-y coordinates of the line segment, respectively. Some of the extracted lines are ignored if their lengths are less than the predefined threshold. Figure 3 shows the results of this line extraction process.

[1] Since we are concern with two-dimensional images, there are two eigenvectors. However, this approach can be easily generalized to n-dimensional data.

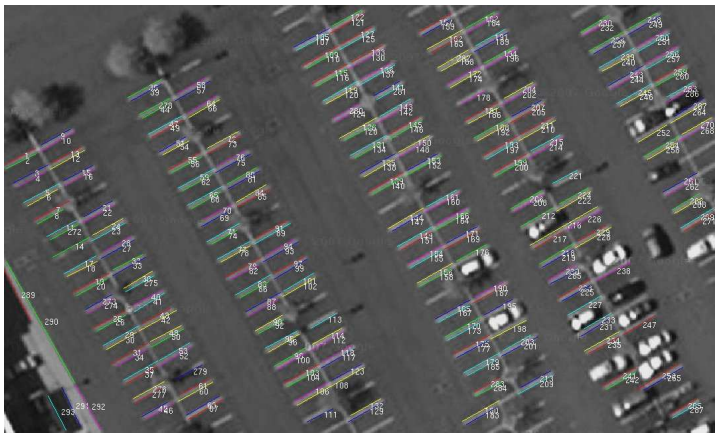


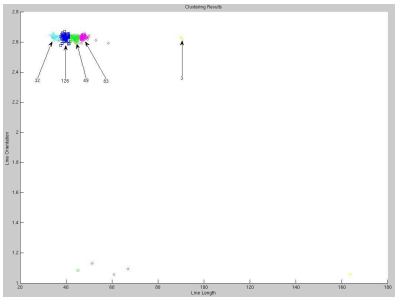
Figure 3: An example of the lines extracted from an image. The parking lot image is captured from the *Google* map service. The magnitude threshold of the image gradient is 30% of the maximum magnitude and line length threshold is 30 in pixels. The random colors are assigned to lines for visualization purpose. There are 293 lines extracted from the aerial image of the parking lot in a shopping mall.

3.1.2 Line Clustering

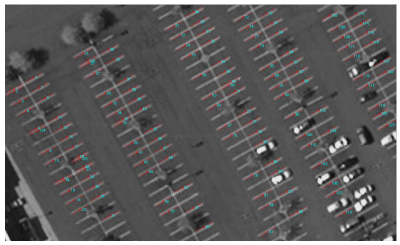
A majority of extracted lines in the figure 3 align with lane markings of the underlying parking lot. However, some of them come from other image regions such as boundaries of the parking lot or contours of other adjacent buildings. Since we only want the lines aligned with the line-markings of the parking lot, it is necessary to remove lines that do not belong to parking lot structure. To this end, we group the extracted lines into clusters where individual clusters have consistent orientation and length. The cluster with the largest member is the one that has most of lines attached to the lane markings of a parking lot. Figure 4 shows a result of this line clustering process. The cluster with the largest member is selected and all of the lines belonging to this cluster are used for parameter estimation. Figure 4(b) shows these selected lines.

3.1.3 Parameter Estimation

The filtered lines from the line clustering process are used to estimate parameters of parking spot. As a first step, we quantize the lengths and compute the mode of the quantized lengths to esitmate the nominal height of parking spot. Next, we build Euclidean distance matrix of all possible line pairs, quantize the distance and compute the mode to obtain the height of parking spots within a lot. Lastly we quantize orientations of lines and compute the mode again to estimate the orientation of parking spots' open-end. Figure 5(a) shows the extracted lines that satisfy these parameters of a parking spot. By using these lines, the low-level image analysis proposes the initial hypotheses on the true parking spots. Figure 5(b) shows the initial parking spots



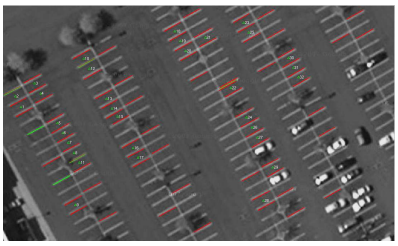
(a) The result of clustering extracted lines.



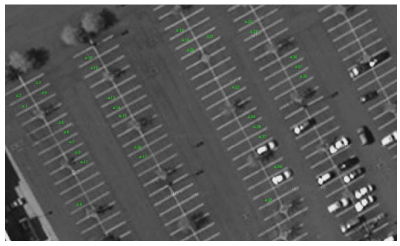
(b) There are 126 lines in the largest cluster.

Figure 4: The extracted lines are grouped into clusters based on their orientations and lengths. The x-axis is line length, $l \in [min(\text{line length}), max(\text{line length})]$ and the y-axis is line orientation, $\theta \in [-\pi, \pi]$. The $min(\text{line length})$ and $max(\text{line length})$ are heuristically chosen to filter out lines that are either longer or shorter than ones that are considered as parts of parking lot structure. To obtain the best result, we apply two well-known clustering algorithms: k-means and Expectation and Maximization (EM) [4]. We find a slight difference in their results and use the simpler and faster one, k-means algorithm. Only the selected lines from the clustering are used for parameter estimation.

hypotheses.



(a) There are 75 lines filtered out again that are satisfied with the estimated parking spots' parameters.

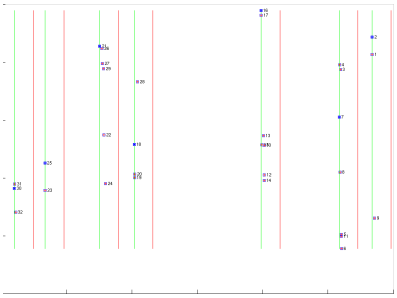


(b) There are 32 initial parking spot hypotheses that are proposed based on the estimated parking spots parameters.

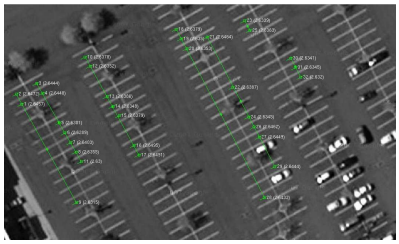
Figure 5: The extracted lines are again filtered out and then used to proposal the initial parking spot hypotheses.

3.1.4 Parking Block Identification

In order to estimate distances between parking blocks, parking blocks must first be generated from the detected parking spots. To this end, we project the centroids of all the intial parking spot hypotheses onto a virtual line whose orientation is the mean



(a) A (green) vertical line represents the beginning of a parking block; the red line indicates the end of a parking block. The distance between peer parking spots (i.e. $D1$ in the Figure 2) is used to determine the distance between the red and green lines. The x-axis is ρ and the y-axis is θ.



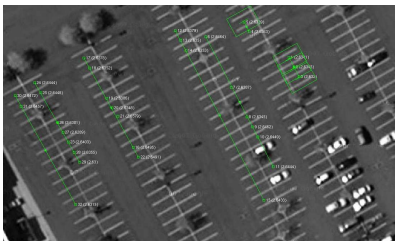
(b) Seven different (green) solid lines over parking spots represent discovered parking blocks.

Figure 6: A heuristic is used to discover parking blocks from the initial parking spot hypotheses.

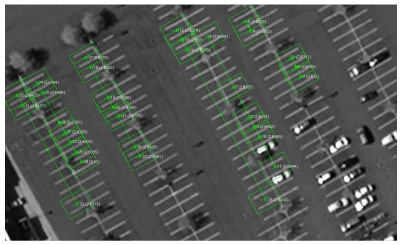
of the initial parking spots' orientation. This projection returns distances of centroids from the origin, ρ.

$$\rho_i = c_{i,x}\cos(\theta_i) + c_{i,y}\sin(\theta_i)$$

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 projections, boundaries between parking blocks are clearly appeared as shown in 6(a). The distance between peer parking spots (i.e. $D1$ in the Figure 2) 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 2).



(a) There are 32 (Green) squares that represent the initial parking spot hypotheses. Rectangular image patches around hypotheses depict candidate parking spots that are evaluated to see if their local image characteristics are similar to those of true parking spots.



(b) Boundaries of the initial candidate parking spots are completely determined.

Figure 7: Rectangular image patches around the initial parking spot candidates are completely determined.

3.2 High-Level Structure Inference

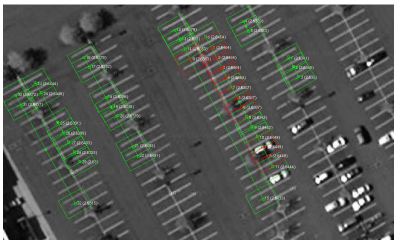
The high-level structure inference is intended to detect all the visible parking spots in an image. To this end, it generates parking spot hypotheses and evaluates rectangular image patches around the hypotheses to evaluate if they are parking spots. A hypothesis represents an image location that represents 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.

3.2.1 Hypothesis Generation

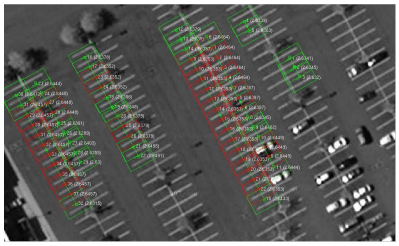
As mentioned earlier, a hypothesis generation process selects image regions that are potentially parking spots based on the initial parking spots detected by the low-level analysis. For each parking spot hypothesis, a rectangular image support region is selected based on the estimated width and height of the parking spot and paddings around the boundaries. Figure 7 shows the progress of this process.

To cover the set of image regions that possibly contain true parking spots, we use the image coordinates of centroids of the initial parking spot hypotheses 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, extrapolation, and block prediction. Figure 8 shows the interpolation process that chooses image regions within a parking block. Figure 9 shows the process of the extrapolation step that investigates outside parking block boundaries to generate parking spot hypotheses.

Figure 9(b) shows the generated parking spot hypotheses through the initial proposal, interpolation, and extrapolation steps. However, as seen in the Figure 9(b), there are three parking blocks that are not covered by the hypothesis generation process. These parking blocks are left undiscovered because there are not any initial parking spot hypotheses proposed in them. To discover these missing parking blocks, we use

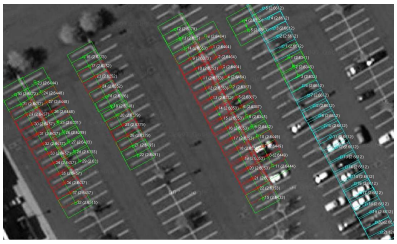


(a) The hypothesis generation begins an interpolation step to cover the empty space within a parking block.

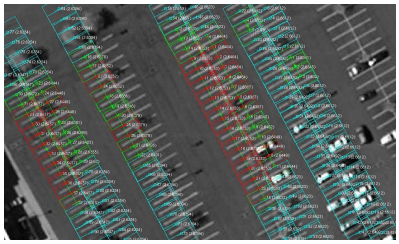


(b) The empty spaces within parking blocks are completely covered by parking spot hypotheses after the interpolation.

Figure 8: The hypothesis generation performs interpolation to generate parking spot hypotheses within a parking block.



(a) The hypothesis generation begins an extrapolation step to cover the empty image regions outside of a parking block.



(b) The empty image spaces outside parking blocks are completely covered by parking spot hypotheses after the extrapolation.

Figure 9: The hypothesis generation performs an extrapolation to generate parking spot hypotheses outside of a parking block boundaries.

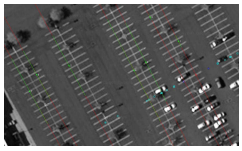
the estimated parking block distances and select image regions to test existence of parking blocks. Figure 10 shows parking block prediction process. Figure 10(c) shows a complete set of parking spot hypotheses.

3.2.2 Hypothesis Evaluation

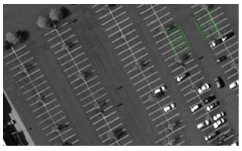
There are n candidate parking spots obtained from the hypothesis generation process, $g_1, g_2, ..., g_n$. Figure 11 shows a complete set of candidate parking spots. Each of the candidate parking spots is evaluated to determine if it is a parking spot. We formulate this decision problem as a binary classification problem for assigning a label, $y_i \in \{-1, +1\}$, to a given image patch (i.e. a candidate parking spot), $\mathbf{g}$.

$$\mathbf{\Omega}(\mathbf{g}_k) = \begin{cases} +1 & \text{if } \mathbf{g}_k \text{ is parking spot} \\ -1 & \text{otherwise} \end{cases}$$

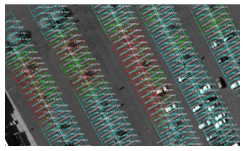
where $\mathbf{g}_k$ is a m-dimensional column vector. In particular, the dimension of the kth candidate parking spot is originally, $\mathbf{g}_k = |\text{height} \times \text{width}|$. It is converted into a m-



(a) Squares show candidate locations for examining parking blocks. The estimated parking block distances are used to generate these locations.



(b) Image regions around each of the squares are examined by projecting the initial parking spot hypotheses onto the candidate region.



(c) We assume that there is a parking block if the similarity between the initial parking spot hypotheses and image regions around the square is greater than the predefined threshold.

Figure 10: The hypothesis generation performs an extrapolation to generate parking spot hypotheses outside of a parking block boundaries.

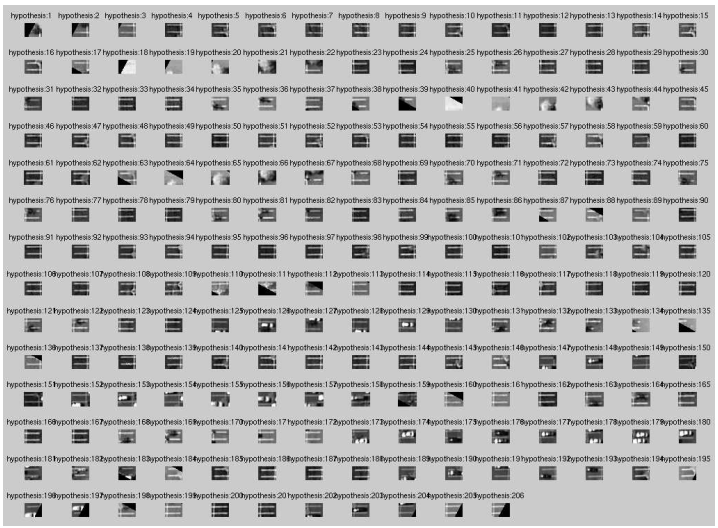


Figure 11: There are 206 candidate parking spots generated through initial proposal, interpolation, extrapolation, and block prediction.

dimensional feature vector, $\mathbf{g}_k = |m \times 1|$ where m is $\mathrm{height} \times \mathrm{width}$ and individual components in a feature vector is an intensity value of a gray image. For the binary classification, we use three different machine learning techniques: Support Vector Machines (SVMs), Eigenspots, and Markov Random Fields (MRFs).

Support Vector Machines SVMs is a supervised learning algorithm that seeks a hyperplane between classes by maximizing margins between them [4]. We choose SVMs as one of the hypothesis evaluation methods because it is easy to use and it is

very useful to handle high-dimensional feature vectors. [2] For training SVMs, we use the initial parking spot hypotheses as positive examples and manually choose the same number of the negative examples from the image. A 10-fold cross validation is used to find the optimal parameters such as the width of Gaussian kernel and the penalty parameter.

Eigenspots Recall that each of the hypothesis evaluation methods needs to examine m-dimensional column vector. Since the number of available candidate parking spots is much less than the dimension of a feature (e.g. m is usually greater than 3,000 in this work), SVMs may not perform well. We need either a sufficient amount of data or an efficient feature representation method to deal with this problem. Since it is not easy to collect a sufficient amount of training data, particularly from a given image, we worked on the development of a better and more efficient feature representation method that represents our high-dimensional feature vector in a low-dimensional space. Turk and Pentland [27] developed a feature representation method called "Eigenfaces." The method utilizes principal component analysis (PCA) [4] to represent high-dimensional human face images in a low-dimension eigenspace. Motivated by this method, in our application, we find the principal components of the initial candidate parking spots (or the eigenvectors of the covariance matrix of the initial candidate parking spots) and treat a parking spot as a point (or vector) in a very high dimensional (parking spot) space. These eigenvectors may be thought of as a set of features that together characterize the variation between the initial candidate parking spots. Each individual parking spot can be represented in terms of a linear combination of the "Eigenspots." In other words, a rectangular image patch of a parking spot hypothesis can be approximated using only the "best" eigenspots, i.e. those that have the largest eigenvalues for capturing most of the variance within the initial candidate parking spots. The best M eigenspots span an M-dimensional subspace of all possible parking spot image patches, where $M << m(m = |g_{\text{height}} \times g_{\text{width}}|)$. From the initial candidate parking spots, the average parking spot is defined by $\Psi = \frac{1}{|\text{initial}|}\sum_{i=1}^{|\text{initial}|} g_i$ and the variance is measured by $\Phi = g_i - \Psi$. In order to find a set of M orthogonal vectors, $\mathbf{u}_M$, which best describe the distribution of parking spot data, we compute the eigenvectors and their corresponding eigenvalues of the covariance matrix, $\Sigma = \Phi^T\Phi$. Figure 12 shows examples of Eigenspots. Each of eigenspots is in fact a new orthonormal basis in parking spot space. To use these eigenspots for the binary classification, we compute the distance between any new image patch, $\mathbf{g}$, and the origin of the eigenspot space. To this end, we need first to compute the variance of a given image patch from the mean parking spot, $\mathbf{g} - \mathbf{\Psi}$ and then project the variance onto these eigenspots, $||\mathbf{E}^T(\mathbf{g} - \mathbf{\Psi})||^2$. Each of the eigenspots has different contributions on the classification decision based on their eigenvalues. Thus the distance needs to be scaled in order to make all the contributions of eigenspots equal, $||\mathbf{D}^{-1/2}\mathbf{E}^T(\mathbf{g} - \mathbf{\Psi})||^2$. For a given new image patch, we compute the distance $T(\mathbf{g})$ from the origin of the eigenspot space:

$$\begin{aligned} T(\mathbf{g}) &= ||\mathbf{D}^{-1/2}\mathbf{E}^T(\mathbf{g} - \mathbf{\Psi})||^2 \\ &= (g - \mathbf{\Psi})^T\mathbf{E}\mathbf{D}^{-1/2}\mathbf{D}^{-1/2}\mathbf{E}^T(g - \mathbf{\Psi}) \end{aligned}$$

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

$$
\begin{aligned}
&= (g-\mathbf{\Psi})^T \mathbf{E}\mathbf{D}^{-1}\mathbf{E}^T(g-\mathbf{\Psi}) \\
&= (g-\mathbf{\Psi})^T \mathbf{\Sigma}^{-1}(g-\mathbf{\Psi})
\end{aligned}
$$

where $\mathbf{D}$ is a diagonal matrix that has the eigenvalues, $\lambda_1, ...\lambda_m$, $\mathbf{E}$ is a matrix of which columns are the eigenvectors of the covariance matrix, and $T(\mathbf{g})$ is in fact the Mahalanobis distance [26] from the origin of the eigenspot space.

Figure 12: Thirty-two eigenspots are obtained.

Markov Random Fields Our first two methods only consider the local characteristics of an image patch to do the binary classification. Thus their performance is limited by the distribution of the training data. That is, their decisions will be inaccurate when the majority of parking spot hypothesis patches are occupied by vehicles (or occluded by other structures) when they are trained with empty parking spot patches. Therefore it is useful to investigate image patches around the patch of interest as well as look at the local characteristics of the image patch. An image patch is highly likely a parking spot when the majority of neighboring patches are parking spots, even the local characteristics of the patch is unlikely classified as a parking spot. To implement this idea, we use a pairwise Markov Random Fields (MRFs) [17]. A pairwise MRFs, $\mathcal{H}$, is an undirected graphical model that factorizes the underlying joint probability distribution, $P(G)$, by a set of pairwise cliques. [3] 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. Specifically a rectangular image patch is modeled as a random variable and their joint probability distribution is factorized by the given undirected graph structure. 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 characterisics of image patches, G. Figure 13 shows the MRFs designed

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

for our task.

$$P(G) = P(G_1 = g_1, ..., G_n = g_n) = \frac{1}{Z}\prod_{i=1}^{N}\Phi(G_i, Y_i)\prod_{j\in N(i)}\Psi(G_i, G_j)$$

where $\Phi(G_i, Y_i)$ is a node potential, $\Psi(G_i, G_j)$ is an edge potential, Z is the partition function. Since these potential functions only measure the compatibility between two random variables, the partition function is needed to ensure that $P(G)$ is a probability distribution. $N(i)$ is neighbor nodes of the ith node where we consider the first-order neighbors.

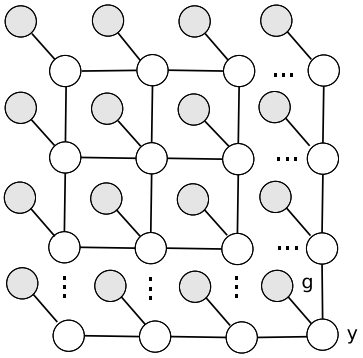


Figure 13: The MRFs used in our work is presented. The empty circles represents unobserved random variables (labels) while the gray circles are observed random variables (image patches).

The MRF is defined through the following potentials:

- The node potential is estimated by a Gaussian Mixture model (GMM) [4] where we assume that candidate parking spots are generated from a mixture of multivariate Gaussian distributions. In order to generate a candidate parking spot, we need first to choose one of the multivariate Gaussian distributions and then generates the patch based on parameters of the chosen Gaussians. In this case, parameters are a mean vector and a covariance matrix. Learning GMM is to estimate parameters of individual Gaussian distributions and their priors from a set of positive and negative examples. [4]. Then the likelihood of a patch given GMM is formulated by

$$\begin{aligned} P(Y = y_i|G_i = \mathbf{g}_i) &= \frac{P(\mathbf{g}_i|y_i)P(y_i)}{P(\mathbf{g}_i)} \\ &= \frac{1}{(2\pi)^{m/2}||\mathbf{\Sigma}_i||^{1/2}}\exp\left[-\frac{1}{2}(\mathbf{g}_i - \mu_i)^T\mathbf{\Sigma}_i^{-1}(\mathbf{g}_i - \mu_i)\right] \end{aligned}$$

[4]To obtain a Gaussian Mixture model, we use GMMBayes available from http://www.it.lut.fi/project/gmmbayes/

We can drop $P(y_i)$ because we have the same number of positive and negative examples.

$$\Phi(G_i, Y_i) = \phi(G_i) = P(Y = y_i|G_i)$$

Since we have two labels, each node has two potentials: a potential being a parking spot, $\Phi(G_i, Y_{+1})$ and the other potential being not a parking spot, $\Phi(G_i, Y_{-1})$.

- The edge potential is modeled by Potts model [17] as follows:

$$\Psi(G_i, G_j) = \psi(G_i, G_j) = exp\left\{-\beta \times (G_i - G_j)^2\right\}$$

where β is a penalty factor for label disagreement between nodes. For example, if $\beta = 0$, edge potentials are identical regardless of the label disagreement and only node potentials are used. The value of an unobserved node is an average of the neighboring nodes and has lots of noise. On the contrary, if $\beta = \infty$, only the edge potentials are meaningful and the node potentials are ignored.

We use the MRFs for estimating the most likely labels of individual parking spot candidates in a given aerial image. To obtain the solution for such inference problem, we use loopy belief propagation because the exact inference is infeasible due to the large number of nodes [12]. We use loopy belief propagation because it is easy to implement and its output is generally reasonable. Loopy belief propagation is a message passing algorithm that estimates marginals, $\mathrm{belief}(G_i)$, over nodes at the convergence. The exchanged messages affect the computation of marginals.

$$\begin{aligned} \mathrm{belief}(G_i) &= \eta\phi(G_i) \prod_{j \in N(i)} \delta_{j \to i}(G_i) \\ \delta_{j \to i}(G_i) &= \psi(G_j)\psi(G_j, G_i) \prod_{k \in N(j)-i} \delta_{k \to j}(G_j) \end{aligned}$$

where η is a normalization constant and $\delta_{j \to i}(G_i)$ is a message from the ith neighbor about what the jth neighbor believes the value of the ith node should be. For the case of undirected graphs with loops, the loopy belief propagation does not always guarantee to converge. If it coverges, it is usually overconfident about the marginals. However, as a solution to the most probable explanation (MPE), it provides reasonable (or at least) useful answers because correct density estimation is not the main concern [6], [36].

4 Experimental Results

The goal of this work is to extract the parking structure from an aerial image. As a first step, we present results for detecting the visible parking spots in an image. We results from three different hypothesis evaluation algorithms and their combination-sSVMs, Eigenspots, MRFs, SVMs with Eigenspots, and MRFs with GMM. Since the performance of these methods are highly dependent on the candidate parking spots generated by the initial proposals and hypothesis generation, we are also interested in investigating how well the hypothesis generation algorithms perform.

(a) Data A: "Dicks." (b) Data B: "Townhouses." (c) Data C: "Dorment."

(d) Data D: "Heinzfield." (e) Data E: "Lowes." (f) Data F: "Costco."

(g) Data G: "Centre Ave."

Figure 14: We use 7 different parking lot aerial images for testing our method. These images are parking lots around Pittsburgh and collected from the *Google* map service.

We collect a number of aerial images from the *Google* [5] map service because it is readily available. For this study, we only use seven of the aerial images for evaluating the usefulness of our methods. Figure 14 shows aerial images used in these tests. There are 1,151 visible parking spots across these different aerial images. Table 1 shows statistics of data.

	A	B	C	D	E	F	G	Sum
Parking Spots	189	84	116	272	222	177	91	1,151
Parking Blocks	10	6	9	10	11	12	4	62

Table 1: Statistics of data.

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

For training SVMs, we use the candidate parking spots initially obtained from the low-level analysis as positive examples. Although they are hypotheses on the true parking spots, we observe that their estimates are quite accurate. We manually generate negative examples from each aerial image. The initial candidate parking spots are also used to build Eigenspots and determine the threshold for the binary decision. For the MRF inference, we build a mesh from the layout of parking spot hypotheses (Refer to Figure 10(c)) 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.

In order to organize outputs of individual methods, we use a 2-way contingency table shown as in Table 2. It summarizes outputs of binary classifiers in terms of four possible outcomes: true positive (*a*), false positive (*b*), false negative (*c*), and true negative (*d*).

		True	
		positive	negative
Output	positive	a	b
	negative	c	d

Table 2: A contingency table is used to measure performances of binary classifiers. The rows of the table are a classifier's outputs and the columns are true labels. Each cell represents one of the four possible outcomes. For example, a is the number of a classifier's outputs that are positive and they are classified as positive (i.e. True-Positives).

We compile this two-way contingency table per output of individual methods and compute the following measures:

- Precision, $p = \frac{a}{a+b}$, if $a + b > 0$, otherwise undefined
- Recall, $r = \frac{a}{a+c}$, if $a + c > 0$, otherwise undefined
- False positive, $fp = \frac{b}{b+d}$, if $b + d > 0$, otherwise undefined
- False negative, $fn = \frac{c}{a+c}$, if $a + c > 0$, otherwise undefined
- Accuracy, $acc = \frac{a+d}{a+b+c+d}$, if $a + d > 0$, otherwise undefined

We first investigate the usefulness of our initial proposals and hypothesis generation. Since each of the hypothesis evaluating methods investigates only the hypotheses that are formed by hypothesis generation and the initial proposals, it is important to know how good they are.

Figure 15 shows the micro-averaged performance of the initial proposal and the hypothesis generation. A micro averaged performance is computed by merging contingency tables over seven different images and then using the merged table to produce performance measures. Surprisingly the initial proposal has zero false positive rate. In

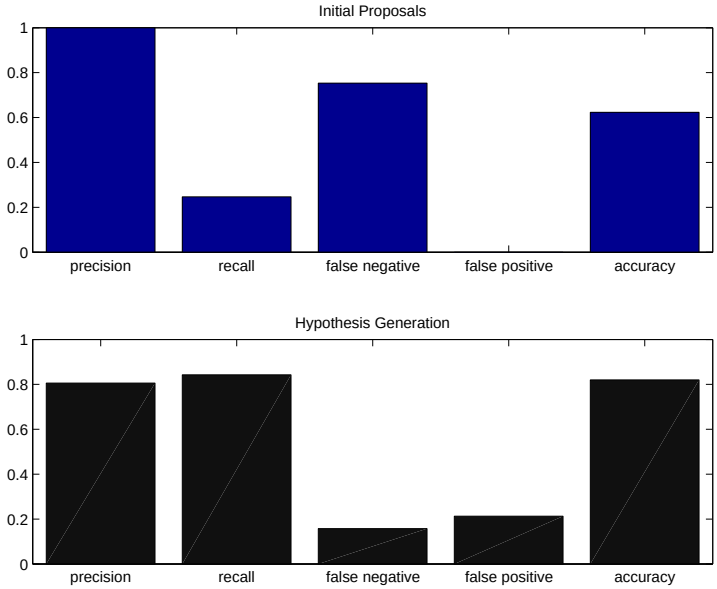


Figure 15: A micro-averaged performance of the initial proposals and the hypothesis generation are shown.

other words, the output of the initial proposals are all true parking spots. The output of the initial proposals would be very useful if they are used directly for a robot navigation without hypothesis evaluation. A false positive output is quite risky for robot navigation. Since a false positive example indicates that there is a parking spot hypothesis classified as a parking spot which is not in fact a parking spot. In the worst case, a false positive output might make a robotic vehicle drive somewhere that the robot should not drive. Despite having no false positives, the initial proposals correctly identifies only 24.67%, on average, of the true parking spots (284 out of 1,151 parking spots.) [6] That is, the initial proposal has a high false negative rate (i.e., 75.33% = 867 = 1,151-284, on average), meaning that 75 percentages of true parking spots are not detected. This may cause another problem for robot navigation: an autonomous robotic vehicle won't be able to park itself even if there are plenty of parking spots available. If the total number of correctly detected parking spots are only 25 percent of all the available parking spots, the robot would conclude that no parking spots are available for the moment. By contrast, hypothesis generation has a relatively low false negative rate, but a high false positive rate. For the hypothesis generation, false positive rate does not really matter because all of the generated hypotheses will be evaluated. However, the false negative rate is important because there is no way to detect the true parking spots if

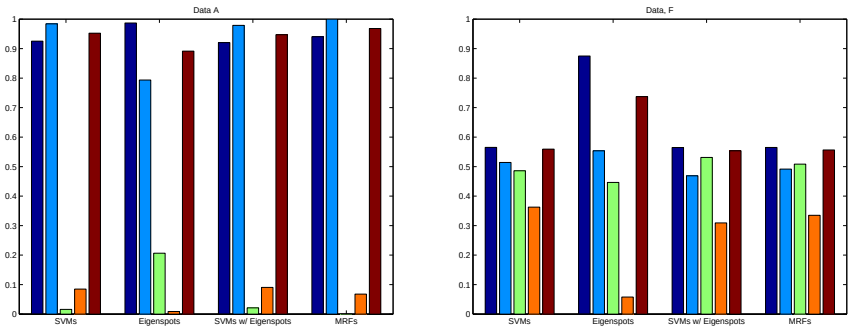
[6]The initial proposals correctly classifies 32 out of 189 for Data A, 49 out of 84 for Data B, 90 out of 116 for Data C, 61 out of 272 for Data D, 27 out of 222 for Data E, 15 out of 177 for Data F, and 10 out of 91 for Data G.

they are not covered by the hypothesis generation. In other words, if the hypothesis generation does not form hypotheses over particular image regions, the true parking spots in those image regions will be eliminated from hypothesis evaluation process and thus not detected. On average, our approach misses 15.73% of true parking spots. To clearly identify the relationship between the performance of the hypothesis generation algorithm and the hypothese evaluation methods, we analyze outputs from two aerial images. Table 3 and 4 show the performance of these stages on two images. In the Table 3, there are 189 visible parking spots in the aerial image A. Although the low-level analysis proposes 32 parking spot hypotheses, it miss 83% of the true parking spots. As hypothesis generation is executed the interpolation through the block prediction, the number of parking spot hypotheses is increased. Accordingly the false negative rate is decreased, but the false positivec rate is increased. At the end of the process, the hypothesis generation covered 100% true parking spots (i.e. zero false negative). This results in the hypothesis evaluation showing good performance (see Figure 16(a)). On the contrary, if the initial proposals and the hypothesis generation do not generate a good set of parking spot hypotheses, the performances of hypothesis evaluation methods are accordingly poor. Table 4 shows that the initial proposal only cover 9% of the true parking spots. This leads to poor parameter estimation of the underlying parking lot. In turn the hypothesis generation fails to discover several parking blocks that increases the false negative rate. The poor parameter estimation also increases the false positive rate. Overall, as shown in Figure 16(b), the performances of the hypothesis evaluation methods are degraded.

Data A ("Dicks")	False Negative	False Positive
Ground Truth (189)		
Initial Proposals (32)	0.83	0.00
Interpolation (37; 69)	0.63	0.00
Extrapolation (90; 159)	0.23	0.06
Block Prediction (47; 206)	0.00	0.09
Hypothesis Generation (206)	0.00	0.09
SVMs	0.01	0.09
Eigenspots	0.20	0.008
SVMs w/ Eigenspots	0.02	0.07
MRFs w/ GMM	0.01	0.06

Table 3: A set of well-defined hypotheses on the true parking spots helps hypothesis evaluation methods improve the performance. The numbers after semicolon represent the accumulated numbers of parking spot hypotheses. For example, after the interpolation process, the sum of the generated hypotheses is 69.

Figure 17 shows the global performance comparisons on seven different images. To measure global performance accurately we use another averaging method: the macro-average which is computed by producing per dataset performance measures first, and then averaging the corresponding measures. While the micro-averaged performance is biased toward the data with the most samples, the macro-averaged one is biased to-



(a) Performance comparison on Data A, "Dicks." (b) Performance comparison on Data F, "Costco."

Figure 16: Performances of the hypothesis evaluaiton methods are dependent on those of the initial proposals and the hypothesis generation.

Data F ("Costco")	False Negative	False Positive
Ground Truth (177)		
Initial Proposals (15)	0.91	0.00
Interpolation (11; 26)	0.85	0.00
Extrapolation (143; 169)	0.56	0.51
Block Prediction (0; 169)	0.56	0.51
Hypothesis Generation (169)	0.56	0.51
SVMs	0.48	0.36
Eigenspots	0.44	0.05
SVMs w/ Eigenspots	0.53	0.30
MRFs w/ GMM	0.50	0.33

Table 4: The performances of the initial proposals and hypothesis generation limit the performance of hypothesis evaluation methods.

ward the dataset with the fewest samples. Under both averaging schemes, Eigenspots perform best in terms of false negative rate while MRFs outperform other methods in terms of the lowest false positive rate. For most of the data, the results follow the same trend with the exception of data, "F." For data "F," since there were not enough number of the initial parking spot hypotheses, all the parking spot hypotheses are poorly generated. That is, most of the rectangular image patches around hypotheses include non-parking spot image regions. This increases high false positive of all the hypothesis evaluation methods. A small number of the initial parking spot hypotheses also increases false negative rate because parking block prediction fails to detect the four missing parking blocks.

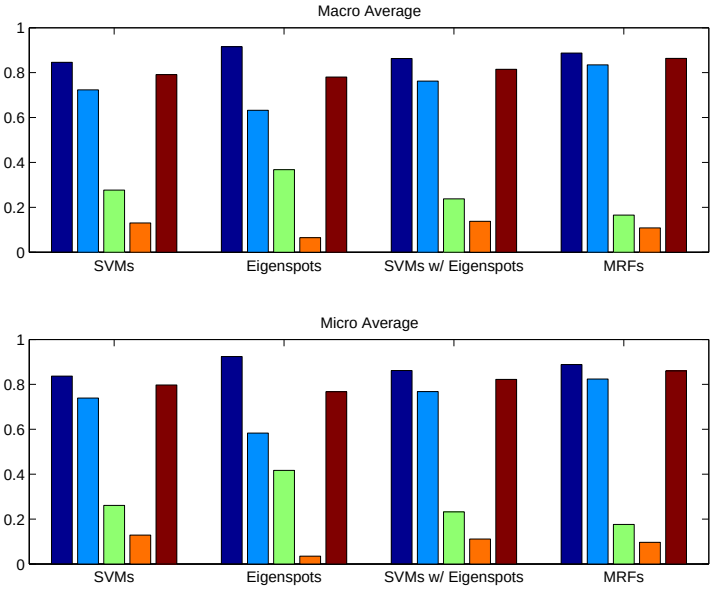


Figure 17: The global performance of four different methods is measured by two averaging schemes. Each group of five bars represents the performance of individual methods. Within a bar group, five performance metrics are (from the left to the right): *precision*, *recall*, *false negative*, *false positive*, and *accuracy*.

5 Conclusions and Future Work

From our experience in the 2007 DARPA Urban Challenge, we believe that the road network information is important for autonomous driving. By providing strong priors on driving environments, this information facilitates perception and planning. However, techniques for coverting imagery into the road network information are labor-intensive, hence error-prone. Therefore it would be very useful to automatically generate the road network information from the given aerial images. As a step toward this goal, we developed an image analysis method that extracts the parking lot structure from a given image.

Since any simple template matching techniques and utilizations of geometric primitives would fail to successfully perform this task due to illumination variation and noisy image acquisition, we approach this task hierarchically. Particularly the low-level aerial image analysis performs low-level image processing procedures to estimate parameters of the underlying structure of parking lot. Based on the estimated parameters, the high-level structure inference extracts the structure of a parking lot by generating and evaluating hypotheses on the true parking spots. Specifically, the high-level structure inference examines rectangular image patches around parking spot hypotheses to determine if they are in fact parking spots. We use three different machine learning

algorithms and their combinations as hypothesis evaluation methods. From the experimental results, we found that the initial proposal and the hypothesis generation limit performances of hypothesis evalution methods. In the current implementation, any hypothesis evaluation methods examine only image regions around parking spot hypotheses. For the generated hypotheses, all the hypothesis evaluation methods showed promising results in that they have less than 10% false positive and less than 17% false negative rate. Particularly Eigenspots showed the best performance in false positive rate whereas MRFs with GMM performed best in terms of false negative rate.

As we pointed out earlier, it is important to generate a set of good parking spot hypotheses. Particularly it is important to have (almost) zero false negative hypothesis generation because the parking spots missed by hypothesis generator will be eliminated by hypothesis evaluation. Our foremost effort on improving the results will be made on improving the performances of the initial proposal and the hypothesis generation. Figure 18 shows three challenging parking lot images. Each of these images poses a unique challenge in that parkings lot in the images have more than one orientation of parking lot lane marking; they contain regions in a high-contrast illumnation change; vehicles in the images occlude substantial parts of images; the quality of lane-markings are poor. Although our methods showed a promising result on extracting parking lot structure, it would not effectively deal with these challenging cases. To effectively handle these cases, we would like to work on the following aspects of our current method.

- To handle variance in illumination, we would like to use spatial filters such as Gabor filter [5] or HOG filter [2]. They have been shown to be robust to changes in illumination and texture in various computer vision applications.

- To handle inconsistent appearances of parking spots, we would like to further examine limitations of Eigenspots and investigate other techniques such as kernel Principal Component Analysis (PCA) or Independent Component Analysis (ICA) [4]. We believe that this would help us generalize representation of parking spots from limited number of initial candidate parking spots.

- To effectively utilize the initial candidate parking spots, we would like to investigate how to assign confidence values on these initial parking spots. Currently we treat them as positive examples of the true parking spots without verifying their qualities. To confidently use them as positive examples, we need to estimate how well they represent true parking spots.

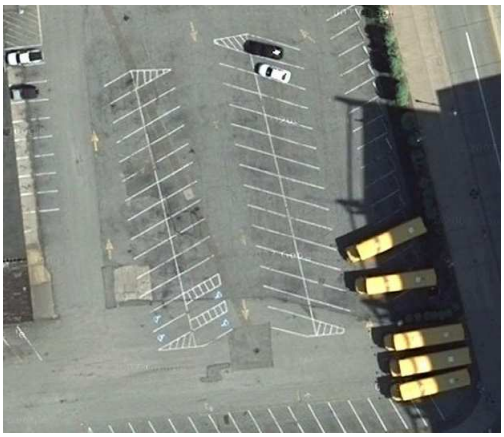
- Using confidence values, we would like to investigate how we approach the task of detecting parking spots in the study of self-supervised learning. A self-supervised learning approach is attractive in the sense that human intervention is not required. To this end, we could generate a set of the negative examples automatically or develop a method that only uses positive examples. One of the ideas to automatically generate negative examples is to utilize the EM algorithm. Particularly, if we provide EM with random image patches from a given image, the ratio of negative examples to the total number of image patches, and the set of positive examples, EM would automatically generate negative examples

from these image patches. One approach for using only positive examples is to utilize the max-margin markov network algorithm [25]. We believe that this algorithm would improve the performance of our method because this algorithm provides us with a way of combining a strength of SVMs (i.e. useful in handling a high-dimensional feature vector) and an advantage of using MRFs (i.e. useful in capturing a global spatial pattern in images.).

- We will work on generating road network information for parking lots given images. To this end, we will begin by identifying boundaries of parking lots.

6 Acknowledgements

This work is funded by GM-Carnegie Mellon Autonomous Driving Collaborative Research Laboratory (CRL). We would like to thank Nathan Ratliff for fruitful discussions on machine learning.



(a) The parking lot image is challenging in that the parking lot has more than one dominant open-end orientations, the image contains regions in high-variant in illumination, and parked buses completely occluded a certain portion of parking lot.

(b) Our current implementation of line extraction won't be effectively able to extract lines from a densely parked image region. Accordingly the number of the initial parking spot hypotheses would be small. This might lead the high-level structure inference to miss a large portion of parking spots.

(c) The current line extraction method might fail to extract a sufficient number of lines from this image because of a poor-quality in lane-markings.

Figure 18: Three challenging parking lot images are shown. Our current method might not effectively handle these three images due to multiple lane-marking orientations, high-contrast illumination, significant occlusions, and poor quality in line-markings.

References

[1] P. Carbonetto, O. D. Freitas, and K. Barnard. A statistical model for general contextual object recognition. In *Proceedings of European Conference on Computer Vision*, pages 350–362, 2004.

[2] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proceedings of Computer Vision and Pattern Recognition*, pages 886–893, 2005.

[3] J. Diebel and S. Thrun. An application of makov random fields to range sensing. In *Proceedings of Neural Information Processing Systems*, pages 291–298, 2005.

[4] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley-Interscience, 2001.

[5] D. A. Forsyth and J. Ponce. *Computer Vision: A Modern Approach*. Prentice Hall, 2003.

[6] W. T. Freeman, E. C. Pasztor, and O. T. Carmichael. Learning low-level vision. *International Journal of Computer Vision*, 40(1):25–47, 2000.

[7] A. C. Gallagher and T. Chen. Using a markov network to recognize people in consumer images. In *Proceedings of International Conference on Image Processing*, pages 489–492, 2007.

[8] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge Univ. Press, 2004.

[9] C.-C. Huang, S.-J. Wang, Y.-J. Chang, and T. 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.

[10] M. I. Jordan. Graphical models. *Statistical Science Special Issues on Bayesian Statistics*, 19:140–155, 2004.

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

[12] D. Koller, N. Friedman, L. Getoor, and B. Taskar. *Introduction to Statistical Relational Learning*, chapter 2: Graphical Models in a Nutshell. MIT Press, 2007.

[13] J. Kosecka and W. Zhang. Efficient computation of vanishing points. In *IEEE International Conference on Robotics and Automation*, pages 223–228, 2002.

[14] J. Kosecka and W. Zhang. Video compass. In *Proceedings of European Conference on Computer Vision*, pages 476–490, 2002.

[15] S. Kumar and M. Hebert. A hierarchical field framework for unified context-based classification. In *Proceedings of International Conference on Computer Vision*, pages 1284–1291, 2005.

[16] Q. Li, N. Zheng, and H. Cheng. An adaptive approach to lane markings detection. In *Proceedings of Intelligent Transportation Systems*, pages 510–514, 2003.

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

[18] C. Lipski, B. Scholz, K. Berger, C. Linz, T. Stich, and M. Magnor. A fast and robust approach to lane marking detection and lane tracking. In *IEEE Symposium on Image Analysis and Interpretation*, pages 57–60, 2008.

[19] A. Mecocci, R. Lodola, and U. Salvatore. Outdoor scenes interpretation suitable for blind people navigation. In *Proceedings of International Conference on Image Processing and Its Applications*, pages 256–260, 1995.

[20] B. Nabbe, S. Kumar, and M. Hebert. Path planning with hallucinated worlds. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3123–3130, 2004.

[21] A. Saudi, J. Teo, H. H.A.Mohd, and J. Sulaiman. Fase lane detection with randomized hough transform. In *Proceedings on International Symposium on Information Technology*, pages 1–5, 2008.

[22] A. Saxena, S. H. Chung, and A. Y. Ng. 3-d depth reconstruction from a single still image. *International Journal of Computer Vision*, 2007.

[23] A. Singhal, J. Luo, and W. Zhu. Probabilistic spatial context models for scene content understanding. In *Proceedings on Computer Vision and Pattern Recognition*, pages 235–241, 2003.

[24] T.-Y. Sun, S.-J. Tsai, and V. Chan. Hsi color model based lane-marking detection. In *Proceedings of the IEEE Intelligent Transportation Systems Conference*, pages 1168–1172, 2006.

[25] B. Taskar, C. Guestrin, and D. Koller. Max-margin markov networks. In *Proceedings of Neural Information Processing Systems*, 2003.

[26] M. E. Tipping and C. M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society*, pages 611–622, 1999.

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

[28] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. N. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer, M. Gittleman, S. Harbaugh, M. Hebert, T. M. Howard, S. Kolski, A. Kelly, M. Likhachev, M. McNaughton, N. Miller, K. Peterson, B. Pilnick, R. Rajkumar, P. Rybski, B. Salesky, Y.-W. Seo, S. Singh,

J. Snider, A. Stentz, W. R. Whittaker, Z. Wolkowicki, J. Ziglar, H. Bae, T. Brown, D. Demitrish, B. Litkouhi, J. Nickolaou, V. Sadekar, W. Zhang, J. Struble, M. Taylor, M. Darms, and D. Ferguson. 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.

[29] J. Verbeek and B. Triggs. Scene segmentation with conditional random fields learned from partially labeled images. In *Proceedings of Neural Information Processing Systems*, 2007.

[30] V. Voisin, M. Avila, B. Emile, S. Begot, and J.-C. Barget. Road markings detection and tracking using hough transform and kalman filter. In *Proceedings of Conference on Advanced Concepts for Intelligent Vision Systems*, pages 76–83, 2005.

[31] X. Wang and A. 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.

[32] Y. Wang, D. Shen, and E. K. Teoh. Lane detection using spline model. *Pattern Recognition Letters*, (9):677–689, 2000.

[33] Y. Wang, E. K. Teoh, and D. Shen. Lane detection and tracking using b-snake. *Image and Vision Computing*, 22:269–280, 2004.

[34] J. Weinman, A. Hanson, and A. McCallum. Sign detection in natural images with conditional random fields. In *Proceedings of International Workshop on Machine Learning for Signal Processing*, pages 548–558, 2004.

[35] Q. Wu, C. Huang, S. yu Wang, W. chen Chiu, and T. Chen. Robust parking space detection considering inter-space correlation. In *Proceedings of IEEE International Conference on Multimedia and Expo*, pages 659–662, 2007.

[36] J. S. Yedidia, W. T. Freeman, and Y. Weiss. *Exploring Artificial Intelligence in the New Millennium*, chapter 8: Understanding belief propagation and its generalizations. Elsevier Science, 2003.

[37] B. Yu and A. K. Jain. Lane boundary detection using a multiresolution hough transform. In *Proceedings of International Conference on Image Processing*, pages 26–29, 1997.