ALBERT-LUDWIGS-
UNIVERSITÄT FREIBURG

Institut für Informatik
Autonome Intelligente Systeme
Prof. Dr. Wolfram Burgard

Where to park?
An in-vehicle parking space occupancy estimation and guidance system

Masterarbeit

Igor Bogoslavskyi
März 2014

Betreuer: PD Dr. Cyrill Stachniss
???????????WHOISHERE??????????????

Eidesstattliche Erklärung

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, keine anderen als die angegebenen Quellen/Hilfsmittel verwendet habe und alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten Schriften entnommen wurden, als solche kenntlich gemacht habe. Darüber hinaus erkläre ich, dass diese Arbeit nicht, auch nicht auszugsweise, bereits für eine andere Prüfung angefertigt wurde.

(Igor Bogoslavskyi)
Freiburg, den 22. Februar 2012

Abstract

As there are more and more cars on the streets from day to day it gets more and more complicated to find a parking space, especially in the center of the city, where one has spacial constraints on building new parking lots.

Therefore it comes in handy to map and predict the occupancy of parking spaces. This information, when integrated into planners commonly found in cell phones and GPS navigation systems, can drastically improve the experience of searching for a parking lot.

In this thesis we present an automated approach of gathering and interpreting the parking lot occupancy information using a mobile platform.

It allows for planning the route not only with respect to spatial information but also taking into account parking lot occupancy information.

The proposed approach uses an in-vehicle camera setup to repeatedly detect and map the parked cars in the area of interest and to estimate the occupancy probability of each found position throughout all runs. The framework is easily expendable to account for searching for occupancy on specific date or even time of day.

Furthermore, we also introduce the planner that relies on the occupancy probability of each found parking lot to find the optimal path in the means of time spent on searching for a parking place as well as getting from the found parking lot to the destination by foot.

Contents

1 Introduction

We all have been in a situation, when one was driving around a city, trying to find a free parking space. One can drive numerous circles around the center of the city with absolute uncertainty about where and when he can eventually find a free parking lot.

Nowadays there are more and more cars in the streets every day, the cities grow and more people use their car to get from place to place, therefore the time spent on the search for the free parking spot takes increasingly more time.

There is a lot done nowadays to make it easier for people to park their cars — most indoor and some outdoor parkings offer a number of free parking lots available at the moment. The modern maps usually contain the positions of all open parkings mapped with their GPS coordinates and can be easily found with an ordinary navigator that one can nowadays find in most cell phones. Even though these occupancy counters make it easier to find a spot, the areas that have no parkings with occupancy information estimate are still common.

Furthermore, it is not always convenient to go to a parking garage as sometimes they lie far from the wanted destination. It is sometimes a lot more suitable to leave a car in one of the on-street parkings.

There are, however, currently no solutions that provide any information on the positions of the parking lots in the streets and their availability. Therefore, we are on our own with this problem. This results in having to spend valuable time on repeatedly driving along the same street in a desperate hope that someone has left and there is a parking spot available now. It is also usually the case that the on-street parkings are not precisely mapped and the navigators we use every day have no notion about them.

This problem is annoying for humans, but people are dealing with such uncertainty much better then the robotized systems.

Recently autonomous vehicles have become a topic of great interest. In the latest years, auto companies and software companies as well as the universities all over the world have focused on developing their own autonomous vehicles.

These vehicles can already impressively well navigate in the cities (Leonard et al. (2009); Lima and Pereira (2013); Thrun et al. (2007)) and are able to take people to

arbitrary point even in hard to analyze urban environments. These cars are becoming safer, they learn how to adopt behavioral knowledge from the traffic (Maye et al. (2011); Spinello et al. (2010)), are more efficient in the means of fuel consumption and a are lot more predictable, causing therefore less traffic jams. We believe, that with time people will spend less time driving by themselves and will rely more on robotized autos.

When it comes to parking, these vehicles can find out what a parking space is and are able to park there (Burgard et al. (2011); Lee et al. (2009); Wang et al. (2011)). Despite being able to park in an autonomous fashion, they are yet unable to know where to search for a free one by themselves and as a consequence where to search for a good one.

We want to address precisely this issue. We present an approach that provides a system for predicting the occupancy in any area as well as an algorithm for on-ground mapping of the parking lots with such additional information. Not only we are able to estimate this information but we also present a way to plan the route in such way that it eliminates uncertainty in finding the free parking place and guarantees to find an optimal parking lot, in the means of the overall time spent for the search, parking and walking to the end destination.

1.1 Contribution of this thesis

We have build a system that combines perception, mapping and planning for solving the problem of finding a free parking spot. The system combines HOG-based visual detection of the parked cars with mapping with occupancy grid maps and an MDP-based planner.

2 Related Works

The problem of finding free parking lots receives more attention with the continuous growth of the cities and their population. As far as we have found out, we can divide the works in the field in two main categories depending on the sensor setup. The detection of the parking lots is either carried out via the usage of static-mounted cameras or the on-vehicle sensor systems. We will further focus more on each of these options.

2.1 Detection with static monocular cameras

To the best of our knowledge, the main purpose of such systems is to simplify the sensor setup for occupancy detection in the modern parking garages. These works differ in the features utilized for car detection as well as in methods of clustering these features.

Qi Wu (2006) presents an unsupervised system to monitor the occupied parking slots. The authors were using a stationary monocular camera to detect parked vehicles. In their work, car detection is solely based on color. The authors use the support vector machines (SVM) to cluster the features to distinguish between occupied and free parking lots. They pre- process the ground truth images of the parking lots and search for color differences between the measurements and the ground truth measured for an empty parking lot.

their or his?

True (2007) shows a similar approach. Alike with the previous papers he presents an unsupervised system for parking lots detection. The author uses an overhead static camera. He uses human-labeled parking lots' positions to define the spacial arrangement of the parking lots. The author distributes chrominance channels of the images from the camera into bins, storing a histogram of these values for each parking lot, then he classifies these histograms as free or occupied using either k-nearest neighbors or SVM. He also presents a variant of the work based on classifying color patches situated around corner detector's regions of interest, but he proves it to be less efficient.

Marc Tschentscher (2012) utilizes a static monocular camera and shows a comparison of different features for occupancy analysis. As in the works mentioned above, the authors propose using an overhead camera pointing to the parking lots, which are manually labeled in the images. They study the influence of the visual features such as color

histograms, gradient histograms, difference of Gaussian histograms and Haar features on the detection performance. They also explore the variance in methods for training a classifier based on the chosen feature set, such as k-nearest neighbors, linear discriminant analysis and SVM. They achieve detection rate of 98% while performing in real time.

Ichihashi et al. (2010) shows a system that uses a set of overhead cameras for occupancy detection in indoor and outdoor scenes. The focus is set on the clustering algorithms, showing that the performance of the detector based on the fuzzy c-means (FCM) clustering and the hyper-parameter tuning by particle swarm optimization (PSO) outperforms SVM both in speed and accuracy of detection.

Huang and Wang (2010) and Huang et al. (2008), use a static monocular overhead camera and a 3-layer Bayesian hierarchical framework (BHF) the authors of the paper specifically addressed the challenges of vacant parking space inference that come from dramatic luminance variations, shadow effect, perspective distortion, and the inter-occlusion among vehicles showing great detection rates of up to 99%.

Some works change the focus from detecting the parked cars to detecting the appearance of the markers pre-painted on each parking lot.

R. Yusnita (2012) proposes a system to detect occupancy status of the parking lots of indoor parkings. They are using an overhead, strictly vertically oriented camera. The parking lots are manually labeled by a marker that states their occupancy state. For each query parking lot the authors produce a binary image which is then analyzed to find if the parking lot is empty or occupied. This process is carried out by comparing the shape of the detection with the shape of the prior that reassembles the circle shape of the marker drawn on the floor of the parking slot. (end)

2.2 In-vehicle detection

The main area of interest for us is the in-vehicle detection of the parking lots. We are specifically interested in the works that present approaches relying on the fusion of sonar/laser data with visual information.

Fintzel et al. (2004) and F. Abad (2013) model free parking lots locally based on the 3D sonar sensor mounted on the car with conjunction with a visual 3D estimation setup based on the tracking of the interest points in the images seen from different angles.

In these papers the authors use the ultrasonic sensor and 3D vision sensor that detects points of interest on the bodies of the parked cars and tracks them to compare their positions from different points of view. The cars are then modeled by vertical planes of

two different orientations that are fit into the 3D data from the sensors. The authors model the empty parking lots by empty spaces between the planes.

Coric and Gruteser (2013) also moves focus from static overhead cameras on the developed parking lot to estimating parking lots for on-street parking. They use an ultra-sonic sensor mounted to the side of the car to estimate the parking possibilities along particular streets. The authors use a straightforward threshold method to distinguish between free and occupied space. They also make use of an idea that the more time a car was observed in a place the more it is likely that the space occupied by this car is a valid parking spot. As a result all measurements were incorporated with the GPS measurements to form a global map of parked cars, which the authors used to detect wrongly parked cars.

he????

Following the focus on the in-vehicle sensor setup, Matthias R. Schmid and Wuensche (2012) utilizes an on-board short-range radar. The measurements from three radars are stored into a 3D occupancy grid, that represents the local surroundings of the car. Free parking lots are then detected on the given grid by analyzing occupancy grid cells on curb and on other parked cars. This provides an estimate of the parking lot in 3D that allows for precise parking.

Suhr et al. (2010) proposes a system that is able to detect parking lots from 3D data acquired from stereo-based multi-view 3D reconstruction. The authors select point correspondences using a de-rotation- based method and mosaic 3D structures by estimating similarity transformation. While relying solely on this information and not using the odometry they are able to achieve reliable results with the detection rate of 90%

Some works, however focus not on car detection, but on detecting the parking slots markings which can prove to be useful when analyzing a free outdoor parking.

A fully automatic system for detection of parking slots' markings is presented by Suhr and Jung (2013). The authors utilize a tree structure performing a bottom-up and a top-down approach in order to classify all parking slots as such and leave out all false detections.

2.3 The contribution of this thesis

In this thesis, following the works in In-vehicle detection, we present a system that is capable of detecting parked cars utilizing the in- vehicle sensor setup, consisting of a stereo camera or a monocular camera fused with laser range finder. We also extend this

functionality by estimating the occupancy probability for each parking lot in a global map and additionally present a planner capable of efficiently finding a route to a free parking lot.

My own contribution???

3 Background

In order to be able to solve the given problem we will have to rely on several well known approaches, which we will briefly introduce in this chapter.

3.1 Occupancy Grids

In this thesis we partly make use of the grid maps to model the environment with respect to modeling the probability of cars to be present in the particular part of the world. Grid maps partition the space into a grid of rectangular cells. Each grid contains information about the corresponding area in the environment. In this thesis we make use of one particular realization of grid maps — the occupancy grid maps. Occupancy grid maps assume that each grid cell is either occupied by an obstacle or free. In our case, each cell therefore stores the probability that the particular cell is occupied by a car. The occupancy grid maps are an efficient approach for representing uncertainty. Grid maps allow for detailed representation of an environment without a predefined feature extractor. As they have the ability to represent occupied space, empty space and unknown (unobserved) space, grid maps are well suited for tasks such as path-planning, obstacle avoidance and exploration. In contrast to most representations based on features, grid maps offer constant time access to cells. However, grid maps suffer from discretization errors.

3.2 Object Detection

In this thesis we make extensive use of the object detection to be able to detect parked cars from visual information. Based of the work of Dalal and Triggs (2005) we make use of the histograms of oriented gradients (HOG) features placed in a dense grid on the query images. The histograms from the training data, are divided into positive and negative groups via support vector machines (SVM), presented by Schölkopf and Smola (2003), as shown in Figure 4.1.

Let us shortly describe how the HOG descriptor is formed. The image is partitioned into 8×8 pixel blocks. In each block we compute the histogram of the gradient orientations. This histogram, after the normalization is invariant to changes in light, small deformations etc.

The HOG/SIFT representation has several advantages. It captures edge or gradient structure that is very characteristic of local shape, and it does so in a local representation with an easily controllable degree of invariance to local geometric and photometric transformations: translations or rotations make little difference if they are much smaller than the local spatial or orientation bin size.

3.3 Markov Decision Processes

Markov decision processes are used to carry out complex decisions in a fully observable environment with a stochastic transition model. We make use of the MDPs to find an optimal way of searching for a free parking lot. A specification of the outcome probabilities for each action in each possible state is called a transition model, we will use $T(s \mid s', a)$ to denote the probability of reaching state s' if and action a is carried out in state s. The transitions in the model are assumed to be Markovian. That is, the probability of reaching s' from s depends only on s and not on the history of earlier states. We also need to define the reward function $R(s, a, s')$, that defines a reward for reaching state s' from state s carrying out action a. Overall, the MDP is defined by three components:

- Initial state: S_0
- Transition Model: $T(s \mid s', a)$
- Reward Function: $R(s, a, s')$

These components allow us to define the utility function, which is a measure of how good the state is in the long run:

$$U^{\pi}(s) = E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t) \mid \pi, s_0 = s\right] \tag{3.1}$$

The optimal action is then chosen using the Maximum Utility Principle, that is, choose: the action that maximizes the expected utility of the subsequent state:

$$\pi^*(s) = \underset{a}{\operatorname{argmax}} \sum_{s'} T(s \mid s', a)U(s') \tag{3.2}$$

Provided, that the utility of the state is the expected sum of discounted rewards from that point onwards, we can define the utility via Bellman Equation:

$$U(s) = R(s) + \gamma \max_{a} \sum_{s'} T(s \mid s', a)U(s') \tag{3.3}$$

In order to solve the MDP there are two common strategies to be used: variable iteration and policy iteration. We will only focus on the second, which is briefly introduced here. The policy iteration algorithm alternates the following two steps, beginning from some initial policy π_0,

- *Policy evaluation:* given a policy π_i, calculate $U_i = U^{\pi_i}$, the utility of each state if π_i, were to be executed.
- *Policy improvement:* Calculate a new MEU policy π_{i+l}, using one-step look- ahead based on U_i (as in Equation 3.2).

The algorithm terminates when the policy improvement step yields no change in the utilities. The given method can be shown to converge in $O(n^3)$ where n is the number of vertices in the graph.

4 Our Approach

4.1 Overview

Unlike the works presented in the related work we are interested in integrating all stages in solving the problem such as perception, mapping model and action planning into one system. We further present an integrated framework for carrying out the task of detecting the parked cars, integrating occupancy information into a model that is suitable for further planning and planning for actions itself, when carrying out a decision of finding an unoccupied parking lot. We will further describe the different steps that we have taken on this road as well as the connections between them.

4.2 Perception

In order to detect the positions of the parked cars we first need to visually recognize the parked cars. To achieve this goal we compared a number of approaches for object detection and classification.

The work of Paul Viola (2001) provides good results in the means of detection. The main drawback of the proposed method is that it suffers from in plane and out of plane rotations and from the illumination changes. For the task of car detection it is important to allow for some deviations as the cars can differ in shape and angle of detection and it may be observed in different lighting conditions. Utilizing the local binary patterns (LBPs) as presented by Trefný and Matas (2010) solves the illumination issues and overall boosts the performance. It is, however, alike to Haar features, also vulnerable to rotation and shape changes. Also, to the extent of our knowledge, LBP-based method becomes slow on the training stage, which is not a problem, but still is an important argument, especially on implementation stage.

Considering the above, we decided to apply the HOG-based detector as presented by Dalal and Triggs (2005). This method is based on the gradient and it therefore deals well with illumination and brightness changes and stays robust to the changes in rotation and shape.

However, whichever algorithm we use, we must make a transition from the image space to the world frame. For each detected car we want to know precisely where it is situated with relation to the agent's current position.

The relative position of the detected car to the camera, along with the global position and orientation of the camera itself, is necessary for modeling the occupancy on the global map.

For this purpose we need to exploit depth information, which we obtain from either the range-finder measurements or from the stereo camera. We will further focus on each of these options more precisely.

4.2.1 HOG Detector

Even though we mostly follow the work of Dalal and Triggs (2005), there are still differences between our approach and the one presented him. These differences are due to the fact that we are interested in detecting cars, while the original paper is on detecting people. We are mainly focusing on detecting front/rear facing cars, but the approach is easily extended onto the side-view as-well. To be able to detect the front and read sides of the cars, we need to switch the size of the hog descriptor window from 128×64 pixels to 128×128 pixels. The idea behind this decision is based on generally square-like shape of the cars, when seen from front or rear. Therefore, the square drawn around the car will contain higher percentage of useful information in comparison to a rectangle.

In the training phase we have considered around 1000 manually chosen 128×128 pixel patches containing cars and around 8000 negative examples. Each HOG descriptor is then unfolded into a point in a multi- dimensional space. These points then have to be clustered into positive and negative ones. We describe the background for this decision in the next section.

4.2.2 SVM Classifier

In order to carry out the decision in the test data, we train a linear SVM classifier on top of all HOG descriptors. A more complex decision boundary could of course also be used, but linear SVM, despite its simplicity, provides reasonable results as can be seen in Section 5.

SVM is a well known and arguably the most popular algorithm in binary clustering, therefore many implementation are readily available. In this work we make extensive use of libSVM by Chang and Lin (2011).

The test data detection is carried out in a cascade fashion as presented by Paul Viola (2001) via the sliding window approach. For each sliding window content we create a HOG descriptor which is then tested against the pre-trained SVM classifier in order to find out on which side of the decision boundary it is. If the current HOG belongs to the area where cars are then the current sliding window is a region of interest and contains a car. Each detection is then stored as a rectangle in the coordinates of the image it belongs to. This ends the visual detection part and allows us to move on to the 3D world.

4.2.3 Depth Information

In this section we will describe in details how the depth information was acquired and combined with the visual detections.

Stereo Camera

In order to find the depth from the stereo camera, we need to first calculate the disparity image from left and right images taken from the video stream. Knowing the internal camera parameters we can then reconstruct the relative position of each pixel of the disparity image. We first find the distance along the camera Z axis: $Z = \frac{fB}{d}$, where f is the focal length of the camera, B is the baseline and d is the disparity. After Z is determined we can focus on finding X and Y coordinates from the ordinary projection equations:

$$X = \frac{uZ}{f} Y = \frac{vZ}{f} \tag{4.1}$$

where u and v are the pixel location in the $2D$ image, X, Y, Z is the $3D$ position in the camera frame. When we have the $3D$ positions for every pixel in the images, we can combine this information with the visual detection part. From the previous part we have a region of interest in the image coordinates, which allows us to accumulate the depth of all pixels that fall into it. We can then analyze all of these values to find the distance to the car, that is contained in this particular region of interest. We considered taking either the mean or the median of the chosen values. Given, that the depth that comes from the stereo camera is quite noisy and that the depth values of the pixels originate on the surface of the car and therefore contain quite a big amount of variance, we picked a median as an option as it is the most resistant statistic, having a breakdown point of 50%: so long as no more than half the data is contaminated, the median will not give an arbitrarily large result.

Laser Range Finder

Even though the stereo camera setup is a lot cheaper and easier to mount, it lacks robustness due to the noise that is present in disparity images and to some degree to the erroneous values that fall into the detected region of interest (such as around the car or in the glass parts of it). We therefore also tested a setup with a laser range finder. We used a EUROPA robot Obelix which has a 270° laser mounted approximately on the human knee level. This setup proves to be a lot more robust in terms of finding the exact $3D$ information about the position of the detected car. However, when using laser we need a different approach because there we only have images from a monocular camera and thus do not have a per-pixel association with the depth field the way we have it using the stereo camera. As the laser mounted on the robot provides us with 270° span it is able to cover approximately all the space related to the image from the camera. We then need to find the beams, that span through the area covered by the image. Given the fact, that the camera is mounted on the same Z axis as the laser this can be done in a straightforward fashion. We are interested in the beams, the numbers of which follows this law:

$$\{beam_n | \forall n : \gamma_{start} < \alpha_{camera} + \alpha_0 + \alpha_n < \gamma_{end}\} \tag{4.2}$$

where α_{camera} is the angle of the camera with respect to the direction of the laser, γ_{start} and γ_{end} are the starting and ending angle of the detection bounding box in the image, α_n is the angle of the n-th beam in the laser frame. All the beam endpoints along with the direction of the beams provide us with consistent $3D$ information. In order to account for possible occlusions or mistakes we take the median of these values as a true position of the detected car. After finding the correct car position we move on to the next part and aggregate occupancy information into a consistent spacio-temporal map.

4.3 Model

In the previous sections we have shown the steps of visual detection of the cars and finding their real $3D$ world position. Whichever path we take in the previous step, now we have a number of real world coordinates, that need to be integrated into a consistent map. In this section we will present two approaches that we have chosen to target this task.

4.3.1 Occupancy Grids

In order to map the parking lots along with the occupancy information we utilize grid maps (see Background: Occupancy Grids). In this case we model each cell's occupancy as a static state Bayes filter.

Let $P(\mathrm{a}^i)$ be an occupancy probability estimate of a cell i in the environment. Considering the probabilistic nature of occupancy of every distinct cell we integrate multiple measurement, taken in different times into one model. Following the work of Moravec and Elfes (1985), we obtain an update rule for $P(a_t \mid z_{1:t})$. First, we apply Bayes' rule and obtain

$$P(a_t \mid z_{1:t}) = \frac{P(z_t \mid a_t, z_{1:t-1})P(a_t \mid z_{1:t-1})}{P(z_t \mid z_{1:t-1})}. \tag{4.3}$$

We then compute the ratio

$$\frac{P(\mathrm{a}^i \mid z_{1:t})}{P(\neg\mathrm{a}^i \mid z_{1:t})} = \frac{P(z_t \mid \mathrm{a}^i, z_{1:t-1})}{P(z_t \mid \neg\mathrm{a}^i, z_{1:t-1})} \frac{P(\mathrm{a}^i \mid z_{1:t-1})}{P(\neg\mathrm{a}^i \mid z_{1:t-1})}. \tag{4.4}$$

Similarly, we obtain

$$\frac{P(\mathrm{a}^i \mid z_t)}{P(\neg\mathrm{a}^i \mid z_t)} = \frac{P(z_t \mid \mathrm{a}^i)}{P(z_t \mid \neg\mathrm{a}^i)} \frac{P(\mathrm{a}^i)}{P(\neg\mathrm{a}^i)},$$

which can be transformed to

$$\frac{P(z_t \mid \mathrm{a}^i)}{P(z_t \mid \neg\mathrm{a}^i)} = \frac{P(\mathrm{a}^i \mid z_t)}{P(\neg\mathrm{a}^i \mid z_t)} \frac{P(\neg\mathrm{a}^i)}{P(\mathrm{a}^i)}. \tag{4.5}$$

Applying the Markov assumption that the current observation is independent of previous observations given we know that a patch contains a parked vehicle gives

$$P(z_t \mid \mathrm{a}^i, z_{1:t-1}) = P(z_t \mid \mathrm{a}^i), \tag{4.6}$$

and utilizing the fact that $P(\neg\mathrm{a}^i) = 1 - P(\mathrm{a}^i)$, we obtain

$$\begin{aligned} &\frac{P(\mathrm{a}^i \mid z_{1:t})}{1 - P(\mathrm{a}^i \mid z_{1:t})} = \\ &\qquad \frac{P(\mathrm{a}^i \mid z_t)}{1 - P(\mathrm{a}^i \mid z_t)} \frac{P(\mathrm{a}^i \mid z_{1:t-1})}{1 - P(\mathrm{a}^i \mid z_{1:t-1})} \frac{1 - P(\mathrm{a}^i)}{P(\mathrm{a}^i)}. \end{aligned} \tag{4.7}$$

This equation can be transformed into the following update formula:

$$\begin{aligned} &P(\mathrm{a}^i \mid z_{1:t}) = \\ &\left[1 + \frac{1 - P(\mathrm{a}^i \mid z_t)}{P(\mathrm{a}^i \mid z_t)} \frac{1 - P(\mathrm{a}^i \mid z_{1:t-1})}{P(\mathrm{a}^i \mid z_{1:t-1})} \frac{P(\mathrm{a}^i)}{1 - P(\mathrm{a}^i)}\right]^{-1} \end{aligned} \tag{4.8}$$

In order to gain efficiency, one can furthermore use the log-odds formulation of Moravec (1988), so that the operations in Eq. (4.8) are realized via addition and subtractions in the log-odds space. We can therefore model the probability of each cell to be occupied by a parked vehicle at each time t. This information can be accumulated at any needed time as well as on different days. In order to fully define the equation given above, we need to define the observation model $P(\mathrm{a}^i \mid z)$

$$P(a^i|z_t) = \begin{cases} 0.45, & \text{if before detection} \\ 0.9, & \text{if cell stores detection} \\ \text{prior}, & \text{if after detection} \end{cases} \tag{4.9}$$

Furthermore, the observation model is defined along the rays, that span from the camera position to the defined $z_{\max}$ and along the defined for the camera field of view. Of course, we have an occupancy grid based world, so there has to be a way to compute the cells that fall into the field of view. In order to carry out this action, we first compute left and right further points. We find the cells that form the end frontier by using the Bresenham algorithm as defined by Bresenham (1965).

Then, for each cell we carry out the same algorithm from camera cell to this query cell. Whenever we encounter a cell, where a car is situated, the next cell is also updated as occupied, while all the ones that come afterwards are not updated as "not visible". This can be seen in a small example in Figure 4.2. This gives us the means of formulating full occupancy grid that can store occupancy information from different days. It is theoretically then possible to move on to planning.

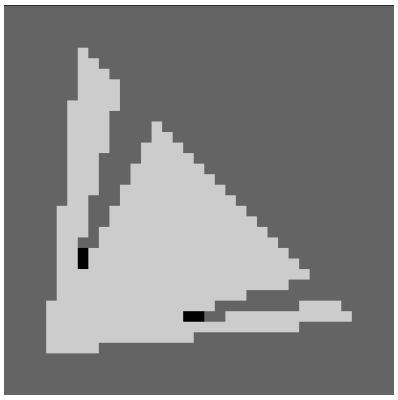

Figure 4.2: An example of Bresenham based update from a single image with two detected cars.

4.3.2 Static State Binary Filter in Pre-defined Positions

However, occupancy grids have their drawbacks. One of them, especially for our setup, is the discretization error. Whenever there are multiple detections of the same car from different angles it can happen, that the detection is assigned to different cells of the occupancy grids. This leads us to the problems of data

association and therefore difficulties in creating a meaningful accumulated map. This, in its turn, leads to wrong decisions of occupancy and ruins or at least makes it harder for a planner to carry out a correct decision. For now we decided to use a pre-defined set of parking lots, that can be set from on-ground data or from aerial images (such as Google Maps). The rest of the theory stays untouched. We still make use of the static state binary Bayes filter as defined in Eq. (4.8). However, the observation model has changed to to the absence of the occupancy grid. We now have different distinct parking lots, that are represented by a point in the space with its coordinates. When we get an observation of occupancy during one full session of measurements, we pick the point to update by using the closest neighbor from the number of available parking lots. The chosen parking lot is updated in a similar way to the occupied case of 4.9. After the full session of observation is carried out, the untouched cells are updated as free. Though this approach does not provide us with a consistent map of the environment we still capture the occupancy information of the correctly situated in space parking lots, which allows us to form a graph that represents the environment and move on to the next part: Action Planning.

4.4 Action Planning

Once we have defined the model for mapping, we are able to formally define the action planning framework that can lead to an optimal decision in the means of picking a path on our way to the parking lot. It depends on the application what an optimal decision is. Whenever we try to find a parking space it is usually the case, that there are parts of the parking lot which are better suited for parking. Usually it means, that these parts are closer to the destination where all the people lean to. This also means that the "better" parts of the parking lots are more occupied than the other, which leads us to a trade-off between going straight to the closest, but likely occupied parking lot, which results in spending time searching for a new one or parking in a more distant area, most likely free, but it takes a while to walk from there to the goal. Given the constraints described above, the framework that we present focuses on finding the solution that minimizes the expected time spent on finding a free parking spot and walking from the found one to the goal.

As stated in the work of Tipaldi and Arras (2011), MDPs can be used when there is a need to maximize the joint rewards instead of greedily going for the best possible goal. We follow similar approach, while defining rewards and transition probabilities based on

the environment that we have in our problem.

In order to solve the given problem in an optimal way, we define rewards, which depict the time spent for each transition from state to state an agent can take. We then use the Markov Decision Processes as described in section 3.3 to find a path that maximizes the reward. The resulting path does not go for the best spot, but maximizes the probability of encountering a free parking lot, while trying keeping the length of the search path as short as possible.

As it is stated in section 3.3, we need to define rewards and actions in order to fully define an MDP problem. First, lets start, by defining the state space. The states are defined as the vertices of the graph V, each of which represents a point in space. The graph also has edges E a function $V \to V$, that correspond to the actions that can be carried out from one state to another. In the grid based situation, the states are the grids cells. In the case, where we explicitly map the parking lots — they, along with their spacial information form the vertices of the graph. In order to define rewards and probabilities of actions we need to define the set of all actions that can be carried out. We are assuming to be in a world, where only five actions are defined — "up": $\uparrow$, "down": $\downarrow$, "left": $\leftarrow$, "right": $\rightarrow$ and "park". Formally:

$$A_{\text{move}} = \{\uparrow, \downarrow, \leftarrow, \rightarrow\} \tag{4.10}$$

$$a_{\text{park}} = \text{“park”} \tag{4.11}$$

$$A = A_{\text{move}} \cup a_{\text{park}} \tag{4.12}$$

Now we can define the reward function $R(s, s', a)$ as well as transition function $P(s|s', a)$. Let us first start with probabilities. For four actions that describe moving around the parking lot in a car we considered the probability of moving from any state to the next one (if possible) to be equal to 1, while the probability of parking in a particular state is equal to the occupancy probability of the state:

$$\forall (s, s') \in E, \forall a \in A_{\text{move}} : P(s|s', a) = 1 \tag{4.13}$$

$$\forall s \in V, (s, s_{\text{goal}}) \in E : P(s|s_{\text{goal}}, a_{\text{park}}) = P(\text{free}) \tag{4.14}$$

where $P(\text{free})$ is a probability of a parking lot to be free, observed through a period of time. To be consistent and to fully define the transaction model, we also need to set the remaining probabilities (the ones not set before) to the values such that $\forall s, s' \in V, (s, s') \in E : \sum_{s'} P(s|s', a) = 1$. This resolves to staying in the same state when trying to carry out an unavailable or improbable action in any state.

For example: it is clear, that staying in the left most corner of the parking lot and

carrying an action to go left, should result in staying in the same spot and while carrying out an action of parking, with the probability of 0.6 has a 0.4 chance to stay in the same place.

Another part of defining the MDP is the definition of the reward function $R(s, s', a)$. The motivation behind the rewards is based on the time it takes an agent to carry out an action. For now we assume, that it takes us a constant time for moving from one state s to another s'. If we cannot carry out the action and are forced to stay in the same position, we anyway lose the time. This describes the fact that each action takes time. We therefore define a negative reward for carrying out any movement:

$$\forall (s, s') \in E, \forall a \in A_{\text{move}} : R(s, s', a) = R(s, s, a) = \text{-const} \tag{4.15}$$

We have defined the costs of moving, which "tells" our agent, that the best strategy would be to stay in one place. However, this is not the behavior we expect from an intelligent agent, so we need to define the rewards of reaching the goal state. The rewards should still be time based, but they have to form a decreasing linear function as the state closer to the destination should get a bigger reward than the one that is situated further. We decided to set the reward to $r_{\max} - r_s$, where $r_s = \sqrt{(s - s_{\text{goal}})(s - s_{\text{goal}})^T}$ is a distance between the arbitrary state $s \in \mathbb{R}^2$ and $r_{\max} = \max_{s \in V} r_s$ is the greatest of those distances. Therefore:

$$\forall s \in V, (s, \text{goal}) \in E : R(s, s_{\text{goal}}, a_{\text{park}}) = r_{\max} - r_s \tag{4.16}$$

This guarantees, that the agent will try to find the way to maximize the reward he gets from parking in the described locations. Combining the defined transition model and the reward function allows us to fully define the MDP, which we are then able to solve with the policy iteration method as described in section Markov Decision Processes. The solution to the MDP guarantees to find an optimal solution to the given problem, which, given the nature of our reward functions is also an optimal solution from the human point of view.

However, the MDPs can sometimes fail on the given problem. The failure can be described as follows: the agent carries out an optimal strategy up to the point when he needs to carry out the parking decision. Let us imagine for a moment that the parking lot at which the agent wants to park is occupied. From the point of view of the MDPs the optimal decision would be to wait and try once again, which is clearly a suboptimal decision. Therefore, we also need to plug in the observations of the world. Whenever we move past some parking lot we check if it is occupied and update the occupancy

probability in this state to 1 or 0 respectively. We are then able to carry out a decision having a better background. To find what the next optimal action would be, we start the policy iteration on the newly defined graph for MDPs, providing the last found policy as initial. Under the assumption, that changing the occupancy probability of the observed vertices does not dramatically change the movement strategy of the agent, this yields optimal planning time because the policy iteration algorithm is carried out to the point when the policy does not change.

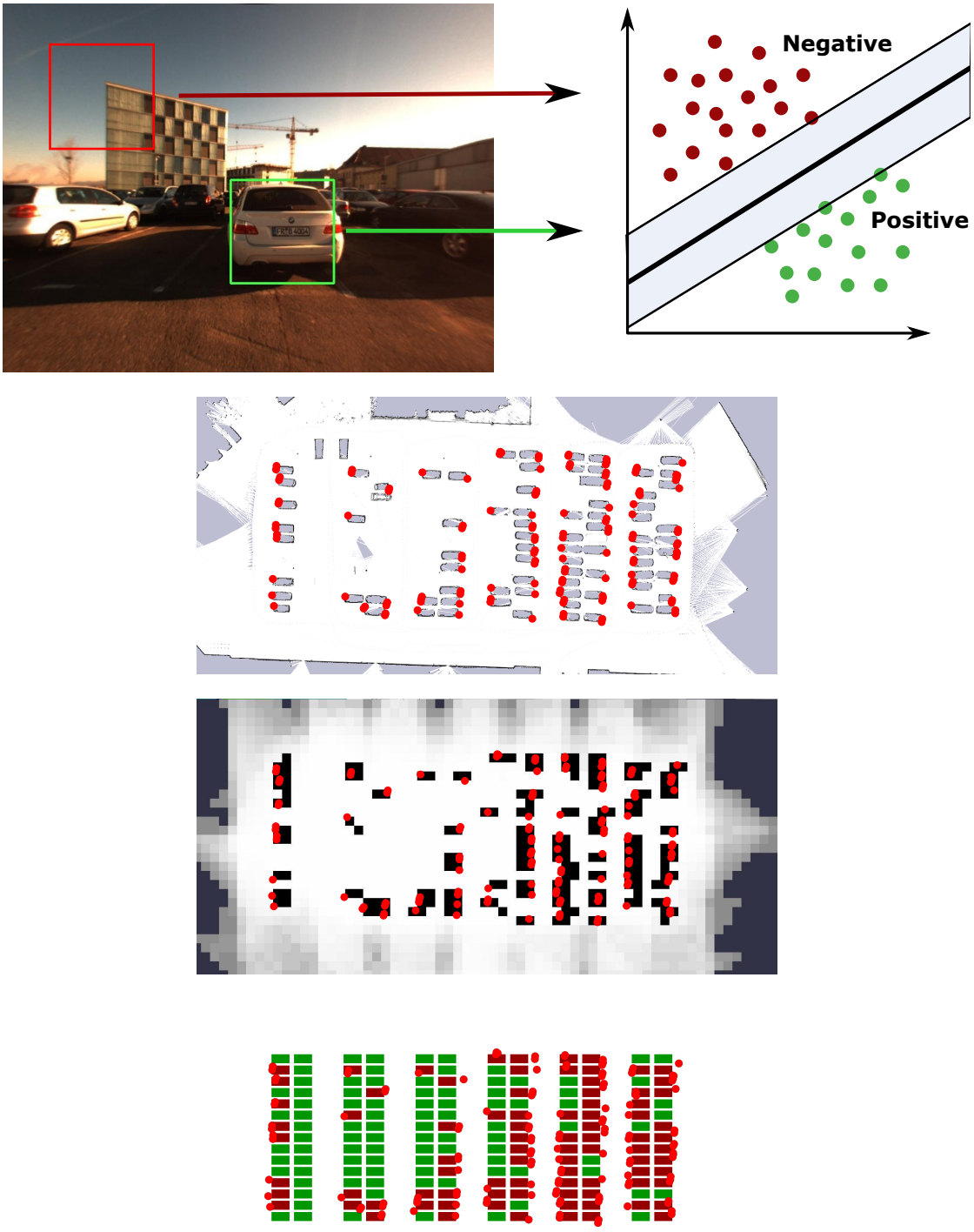

Figure 4.1: Illustration of HOG-based detection and clustering with SVM.

5 Experimental Results

Throughout the course of development of the full framework that we present in this thesis, we have made extensive use of such three components:

- visual car detection
- environment modeling
- planning

Let's focus on each of these as parts of an aggregated system.

5.1 Visual Car Detection

Following the work of Dalal and Triggs (2005), we created a training dataset that contains positive and negative examples that can be fully described with a HOG descriptor in order to cluster the outcome via SVM later. The cars don't look the same from different views, wherefore we have positive training data for different angles of the cars. We have considered to create training samples for eight different views of the cars. However, for our purposes there are some simplifications that can be taken into account. We do not need the diagonal views as the cars are usually observed from the side, front or back. The sides of the car are actually the same if we allow mirror transformation. Front and back of the car, though different to human point of view, seem to be quite similar in the HOG visualization. Therefore, the corresponding datasets can be combined into one. The positive dataset was filled from different sources, INRIA Car Dataset by Carbonetto and Dorkó (2004), Motorway Car Dataset by Caraffi et al. (2012) and images found in a public domain via Google. In this work we were mainly focusing on the front/rear detection. The side detection can be handled in a similar way. The final positive training dataset contains 440 images, 128×128px each. Each of those images contains only a car and can be fully described with a HOG descriptor. The negative set is created in the way described in the work of Dalal and Triggs (2005). First, a number of 128×128px patches

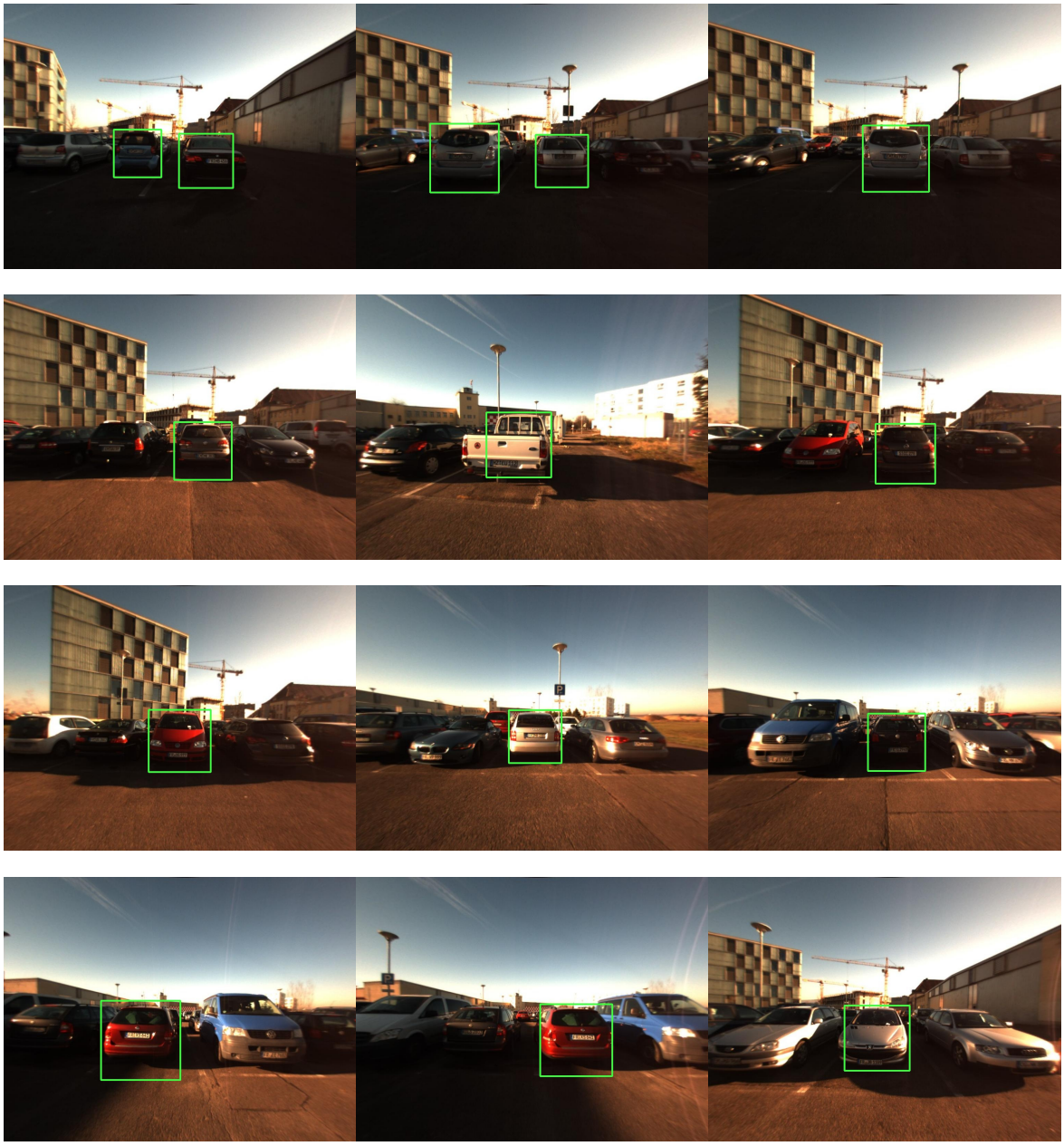

Figure 5.1: Examples of detections of different cars under different light conditions.

is sampled from the images, that contain no cars. This data is then used as the negative data for the classifier training process. The false detections of the outcome are also put to the same negative training set. This is performed several times. The final negative dataset contains 7972 images, each of which is 128×128px big. The system shows nice performance in detecting the cars it faces (we explicitly did not focus on detecting all cars in the scene, such as those, that are found on the side of the image or parts of cars as it makes it harder to find the corresponding 2D position afterwards) and the detection rate of . Some of the detection examples are presented in Figure 5.1.

add detection rate for my approach

5.2 Occupancy Modeling

do a photo of the robot with a camera

We here present the examples of occupancy data aggregated to the occupancy grids as well as into the pre-defined parking lot positions. The occupancy data is based on the 2D data obtained with lasers, mounted on the robot. As can be seen in the figure, the stereo camera is mounted on the same z-axis as the laser range finder. We search for the needed endpoints that represent the cars as described in chapter Our Approach, section Perception The precision of car position modeling can be seen in the figure.

what to write more???

5.3 Planning

As can be seen in the previous section, the question of how to use the occupancy grids in order to build a planner on top of the occupancy information in them is still open. We thus currently focus on a planner based upon the pre-defined parking lots positions. However, the planner itself is generic, which allows to use it in any situation alike. For now we can manipulate the constant time for taking the next action. It can be seen, that if this value is set to a very big value, the optimal action for each state is to park right there just to keep the overall reward positive. If we on the contrary set this constant to 0, the optimal decision will be to move around the parking lot until we have found the best (closest to the goal) parking lot and then try to park there. The most optimal result is achieved if the cost of moving between the parking lots by car is the distance between them, divided by the speed of the car. This makes the algorithms consider a trade-off between shorter routes while searching for a place and better parking lots, situated closer to the goal, weighted by the occupancy probability. The next example shows the behavior of the agent under different current occupancy.

6 Conclusion and Further Work

We have build a system that integrate perception, mapping and planning for solving a task of efficiently finding a free on-street parking lot.

The algorithm presented in this work relies on visual detection of the cars in the streets. As no reliable fast car detector was found we have written one, which is going to be made open-source in the near future.

Throughout the course of the work we have experimented with different methods for combining metric information with visual detections such as depth from stereo cameras or lasers. Currently the laser range scanners have proven to provide a better result, however it is of great interest to build a system that would provide the same level of detail using purely visual information.

It is important to have a reliable depth estimate only in case of having a good estimate of the position in the world. We use SLAM (Kretzschmar et al. (2011, 2010); Kümmerle et al. (2011)) for this purpose.

Not only we have experimented with perception, but also with mapping. We have implemented the the occupancy grids based approach, which however suffers from the discretization errors as well as from ad-hoc estimation of the position (including orientation) of the detected cars in the occupancy grid. Eventually we stick to the system where we provide our system with additional knowledge about where the parking lots are situated. This information has to be obtained only once for a new environment.

This system then provides a good estimate of the occupancy information that is received via multiple observations of the same spot on different days or simply in different times. It may be subject for future work to estimate the occupancy information for particular day of week or particular time of a working day, etc. The occupancy information is basically a probability of a parking lot to be free when observed.

This occupancy information along with the position of the parking lots in the world allows for a planner to be introduced. This planner, unlike greedy A* and alike searches not for the best parking lot in the means of position or occupancy, but takes into account the trade-off between these two quantities, minimizing the expected time it takes us to park a car and get to the final goal by foot. It is still possible to experience

problems finding the best parking lot, but our approach eliminates the high uncertainty of this process and provides an opportunity to carry out the parking process in a fully-autonomous fashion, while guaranteeing to find the best action in the means of expected time needed for the whole process.

There is however place for future work. To see a fully integrated and easily accessible system we will further focus on making the detection of the parked car position purely based on visual data (via improving stereo-camera depth acquisition). The other vector of interest is improving the approach for mapping in order to leave pre-defined parking lots' positions behind and to be able to efficiently map any given environment without any prior knowledge about it.

Bibliography

J.E. Bresenham. Algorithm for computer control of a digital plotter. *IBM Systems Journal*, pages 25–30, 1965.

Wolfram Burgard, Dieter Fox, and Sebastian Thrun. Probabilistic state estimation techniques for autonomous and decision support systems. *Informatik-Spektrum*, 34 (5):455–461, 2011. ISSN 0170-6012. doi: 10.1007/s00287-011-0561-8. URL `http://dx.doi.org/10.1007/s00287-011-0561-8`.

Claudio Caraffi, Tomas Vojir, Jura Trefny, Jan Sochman, and Jiri Matas. A System for Real-time Detection and Tracking of Vehicles from a Single Car-mounted Camera. In *ITS Conference*, pages 975–982, Sep. 2012.

Peter Carbonetto and Gyuri Dorkó. Inria car dataset, 2004.

Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Vladimir Coric and Marco Gruteser. Crowdsensing maps of on-street parking spaces. In *Proceedings of the 2013 IEEE International Conference on Distributed Computing in Sensor Systems*, DCOSS '13, pages 115–122, Washington, DC, USA, 2013. IEEE Computer Society. ISBN 978-0-7695-5041-1. doi: 10.1109/DCOSS.2013.15. URL `http://dx.doi.org/10.1109/DCOSS.2013.15`.

Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. *Computer Vision and Pattern Recognition*, 2005.

S. Wybo S. Bougnoux C. Vestri T. Kakinami F. Abad, R. Bendahan. Parking space detection. 2013.

K. Fintzel, R. Bendahan, C. Vestri, S. Bougnoux, and T. Kakinami. 3d parking assistant system. In *Intelligent Vehicles Symposium, 2004 IEEE*, pages 881–886, June 2004. doi: 10.1109/IVS.2004.1336501.

Ching-Chun Huang and Sheng-Jyh Wang. A hierarchical bayesian generation framework for vacant parking space detection. *Circuits and Systems for Video Technology, IEEE Transactions on*, 20(12):1770–1785, Dec 2010. ISSN 1051-8215. doi: 10.1109/TCSVT.2010.2087510.

Ching-Chun Huang, Sheng-Jyh Wang, Yao-Jen Chang, and Tsuhan Chen. A bayesian hierarchical detection framework for parking space detection. In *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*, pages 2097–2100, March 2008. doi: 10.1109/ICASSP.2008.4518055.

H. Ichihashi, T. Katada, M. Fujiyoshi, A. Notsu, and K. Honda. Improvement in the performance of camera based vehicle detector for parking lot. In *Fuzzy Systems (FUZZ), 2010 IEEE International Conference on*, pages 1–7, July 2010. doi: 10.1109/FUZZY.2010.5584554.

H. Kretzschmar, C. Stachniss, and G. Grisetti. Pose graph compression for laser-based SLAM. In *Proc. of the Int. Symposium of Robotics Research (ISRR)*, Flagstaff, AZ, USA, 2011. URL `http://www.informatik.uni-freiburg.de/~stachnis/pdf/stachniss11isrr.pdf`. Invited presentation.

Henrik Kretzschmar, Giorgio Grisetti, and Cyrill Stachniss. Lifelong map learning for graph-based SLAM in static environments. *KI – Künstliche Intelligenz*, 24:199–206, 2010. doi: 10.1007/s13218-010-0034-2.

R. Kümmerle, B. Steder, C. Dornhege, A. Kleiner, G. Grisetti, and W. Burgard. Large scale graph-based SLAM using aerial images as prior information. *Journal of Autonomous Robots*, 30(1):25–39, 2011. doi: 10.1007/s10514-010-9204-1.

Chen-Kui Lee, Chun-Liang Lin, and Bing-Min Shiu. Autonomous vehicle parking using hybrid artificial intelligent approach. *Journal of Intelligent and Robotic Systems*, 56 (3):319–343, 2009. ISSN 0921-0296. doi: 10.1007/s10846-009-9319-9. URL `http://dx.doi.org/10.1007/s10846-009-9319-9`.

John Leonard, Jonathan How, Seth Teller, Mitch Berger, Stefan Campbell, Gaston Fiore, Luke Fletcher, Emilio Frazzoli, Albert Huang, Sertac Karaman, Olivier Koch, Yoshiaki Kuwata, David Moore, Edwin Olson, Steve Peters, Justin Teo, Robert Truax, Matthew Walter, David Barrett, Alexander Epstein, Keoni Maheloni, Katy Moyer, Troy Jones, Ryan Buckley, Matthew Antone, Robert Galejs, Siddhartha Krishnamurthy, and Jonathan Williams. A perception-driven autonomous urban vehicle. In Martin

Buehler, Karl Iagnemma, and Sanjiv Singh, editors, *The DARPA Urban Challenge*, volume 56 of *Springer Tracts in Advanced Robotics*, pages 163–230. Springer Berlin Heidelberg, 2009. ISBN 978-3-642-03990-4. doi: 10.1007/978-3-642-03991-1_5. URL http://dx.doi.org/10.1007/978-3-642-03991-1_5.

DaniloAlves Lima and GuilhermeAugustoSilva Pereira. Navigation of an autonomous car using vector fields and the dynamic window approach. *Journal of Control, Automation and Electrical Systems*, 24(1-2):106–116, 2013. ISSN 2195-3880. doi: 10.1007/s40313-013-0006-5. URL http://dx.doi.org/10.1007/s40313-013-0006-5.

Marcel Neuhause Marc Tschentscher. Video-based parking space detection. 2012.

J. Dickmann F. von Hundelshausen Matthias R. Schmid, S. Ates and H.-J. Wuensche. Parking space detection with hierarchical dynamic occupancy grids. *IEEE Intelligent Vehicles Symposium*, 2012.

J Maye, R Triebel, L Spinello, and R Siegwart. Bayesian on-line learning of driving behaviors. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

Hans P. Moravec. Sensor fusion in certainty grids for mobile robots. *AI Magazine*, pages 61–74, 1988.

H.P. Moravec and A. Elfes. High resolution maps from wide angle sonar. In *Robotics and Automation. Proceedings. 1985 IEEE International Conference on*, volume 2, pages 116–121, Mar 1985. doi: 10.1109/ROBOT.1985.1087316.

Michael Jones Paul Viola. Rapid object detection using a boosted cascade of simple features. *Conference on Computer Vision and Pattern Recognition*, 2001.

Yi Zhang Qi Wu. Parking lots space detection. 2006.

Norazwinawati Basharuddin R. Yusnita, Fariza Norbaya. Intelligent parking space detection system based on image processing. *International Journal of Innovation, Management and Technology*, 3rd, 2012.

Bernhard Schölkopf and AlexanderJ. Smola. A short introduction to learning with kernels. In Shahar Mendelson and AlexanderJ. Smola, editors, *Advanced Lectures on Machine Learning*, volume 2600 of *Lecture Notes in Computer Science*, pages 41–64. Springer Berlin Heidelberg, 2003. ISBN 978-3-540-00529-2. doi: 10.1007/3-540-36434-X_2. URL http://dx.doi.org/10.1007/3-540-36434-X_2.

L. Spinello, R. Triebel, and R. Siegwart. Multiclass multimodal detection and tracking in urban environments. *Int. Journal on Robotics Research (IJRR)*, 2010.

Jae Kyu Suhr and Ho Gi Jung. Full-automatic recognition of various parking slot markings using a hierarchical tree structure. *Optical Engineering*, 2013.

Jae Kyu Suhr, Ho Gi Jung, Kwanghyuk Bae, and Jaihie Kim. Automatic free parking space detection by using motion stereo-based 3d reconstruction. *Machine Vision and Applications*, 2010.

Sebastian Thrun, Mike Montemerlo, Hendrik Dahlkamp, David Stavens, Andrei Aron, James Diebel, Philip Fong, John Gale, Morgan Halpenny, Gabriel Hoffmann, Kenny Lau, Celia Oakley, Mark Palatucci, Vaughan Pratt, Pascal Stang, Sven Strohband, Cedric Dupont, Lars-Erik Jendrossek, Christian Koelen, Charles Markey, Carlo Rummel, Joe Niekerk, Eric Jensen, Philippe Alessandrini, Gary Bradski, Bob Davies, Scott Ettinger, Adrian Kaehler, Ara Nefian, and Pamela Mahoney. Stanley: The robot that won the darpa grand challenge. In Martin Buehler, Karl Iagnemma, and Sanjiv Singh, editors, *The 2005 DARPA Grand Challenge*, volume 36 of *Springer Tracts in Advanced Robotics*, pages 1–43. Springer Berlin Heidelberg, 2007. ISBN 978-3-540-73428-4. doi: 10.1007/978-3-540-73429-1_1. URL `http://dx.doi.org/10.1007/978-3-540-73429-1_1`.

Gian Diego Tipaldi and Kai O. Arras. I want my coffee hot! learning to find people under spatio-temporal constraints. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, 2011. URL `http://www.informatik.uni-freiburg.de/~tipaldi/papers/tipaldiICRA11.pdf`.

Jiří Trefný and Jiří Matas. Extended set of local binary patterns for rapid object detection. *Computer Vision Winter Workshop*, 2010.

Nicholas True. Vacant parking space detection in static images. 2007.

Zhao-Jian Wang, Jian-Wei Zhang, Ying-Ling Huang, Hui Zhang, and AryanSaadat Mehr. Application of fuzzy logic for autonomous bay parking of automobiles. *International Journal of Automation and Computing*, 8(4):445–451, 2011. ISSN 1476-8186. doi: 10.1007/s11633-011-0602-4. URL `http://dx.doi.org/10.1007/s11633-011-0602-4`.