Where to Park?
Minimizing the Expected Time to Find a Parking Space

Igor Bogoslavskyi   Luciano Spinello   Wolfram Burgard   Cyrill Stachniss

***Abstract*— Quickly finding a free parking spot that is close to a desired target location can be a difficult task. This holds for human drivers and autonomous cars alike. In this paper, we investigate the problem of predicting the occupancy of parking spaces and exploiting this information during route planning. We propose an MDP-based planner that considers route information as well as the occupancy probabilities of parking spaces to compute the path that minimizes the expected total time for finding an unoccupied parking space and for walking from the parking location to the target destination. We evaluated our system on real world data gathered over several days in a real parking lot. We furthermore compare our approach to three parking strategies and show that our method outperforms the alternative behaviors.**

I. Introduction

Finding a free space for parking a car in a densely populated area is a bothering task for drivers. Autonomous cars will potentially face a similar challenge when driving us through a city: where should they go to maximize the chance to find a free parking space close to the our desired destination—this is also important as nearby parking reduces the time needed to pick us up when we want leave. Rodrigue et. al. [15] argue that people typically spend up to 20 minutes looking for a free parking space in the center of a modern city with more than one million inhabitants and that this accounts for more than 10% of the local traffic.

There are several aids for supporting drivers in finding parking spaces. For example, modern garages often provide information on the garage's occupancy status by offering a counter, that shows the number of free parking spaces available at the moment. Even though these occupancy counters make it easier to find garage with a free parking spot, they only provide momentary occupancy information and are typically only available for large parking garages. Thus, people often tend to leave their cars in a curb parking space closer to the target destination.

There are few solutions that provide occupancy information of curb parking spaces but they typically do not address the planning problem of how to maximize the chance to find one of those free spaces. Examples are the "Parking on demand" system by Pierce and Shoup [13], [14] that adapts the price of curb parking spaces in relation to current occupancy and provides live information on occupancy and pricing and "ParkNet" by Mathur et. al. [11] that estimates

Igor Bogoslavskyi and Cyrill Stachniss are with Institute for Geodesy and Geoinformation, University of Bonn, Germany. Luciano Spinello and Wolfram Burgard are with Institute of Computer Science, University of Freiburg, Germany.

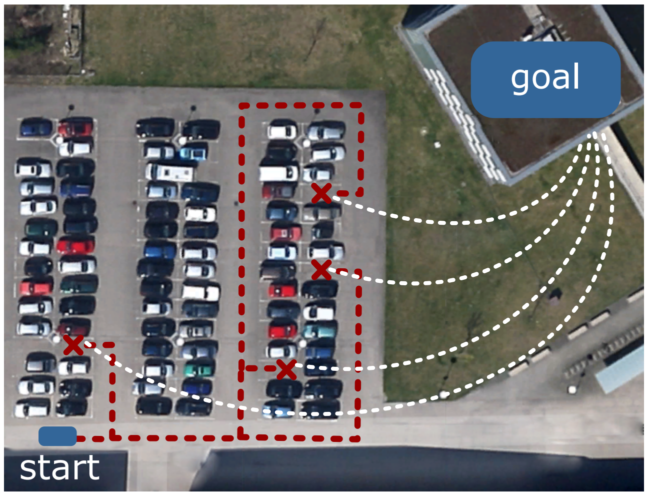


Fig. 1: Where is a free space for parking? The state of the parking lot is not known and the agent wants to spend a minimum amount of time for finding a free parking space and walking to the goal location. This is the problem addressed in this paper.

the positions of the parking spaces with sensors mounted to city taxis.

This paper aims at going a step further and proposes a planning system that guides a car given uncertain occupancy information so as to maximize the chance of finding a free parking spot quickly. Note that this is a different problem than the autonomous parking function of modern cars. Such systems park autonomously *given* a free parking spot [9]. The task of how to navigate through the environment to effectively find one, however, is the question that we are addressing in this work. We address the problem by using an MDP-based planner that considers route information as well as the occupancy probabilities of the parking spaces. Using an MDP to model our problem is the natural choice as it allows us to take into account the probabilistic nature of the parking lot occupancy. We use policy iteration to compute a solution to the MDP which allows for efficient replanning if individual parking spaces change their occupancy based on new observations.

II. Related Work

Most of the work in the area of parking has been conducted for detecting other cars and parking spaces, while a comparably small number of papers focusing on planning routes to find a parking space. The most prevalent systems for providing parking spaces occupancy information utilize overhead cameras to survey the parking lot and infer the positions of the parked cars and respectively free spaces through range sensors or cameras, for example, [23], [21], [22], [7], [8]. These approaches mainly rely on manually labeled parking spaces and focus on detecting the occupancy status by classification using different visual features. Wu and Zhang [23] use color features while True [21] uses histograms over chrominance channels of the images and color patches around the corner detector's region of interest. Tschentscher and Neuhausen [22] compare the performance based on the choice of different visual features: color histograms, gradient histograms, difference of Gaussian histograms and Haar features. On the classification side most of the papers make use of support vector machines classifier [23], [21], [22], [8], some of them focusing on comparing SVMs to other methods, like k-nearest neighbors [22] or fuzzy c-means [8].

Fewer works address the problem of in-vehicle parking space detection. Coric and Gruteser [3] estimate the positions of parking spaces for on-street parking. They use an ultrasonic sensor mounted on the side of cars to estimate the parking possibilities along particular streets using a thresholding method to distinguish between free and occupied space. All measurements are incorporated with the GPS measurements to form a global map of parked cars. Following the focus on the in-vehicle sensor setup, Schmid et. al. [17] utilize on-board short-range radars. The measurements from three radars are stored into a 3D occupancy grid, that represents the local surroundings of the car. Free parking spaces are detected in 3D on the given grid. Likewise Suhr et. al. [19] propose a system that is able to detect parking spaces from 3D data acquired from stereo-based multi-view 3D reconstruction. Looking at the problem from a different perspective, Suhr and Jung [18] present a fully automatic system for detecting the markings of the parking spaces.

Autonomously parking a car *given* a free parking space is available today in new cars, whereas the problem of how to navigate in order to find a free parking space has not received a lot of attention in our community. Different authors address the problem of how to navigate in a parking lot, for example, for generating paths for an autonomous vehicle in unknown environments given the destination parking space [5], [1]. This is, however, a substantially different problem and both works assume to know the location of the free parking space that the vehicle should reach. In contrast to that, we aim at computing the path through the environment to minimize the expected time to park the car and reach the target location.

III. Car Detection and Parking Lot Modeling

Our planning approach for effective parking relies on the ability to detect cars. For this paper, we used a combination of an onboard camera and a 2D laser range finder. Please note that our planning approach presented in Sec. IV is independent from the detection approach itself—various alternative methods for car detection can be used without modification of the remaining parts.

A. Car Detection

We detect cars based on visual features extracted from a Bumblebee stereo image stream and compute the positions of the detected cars relative to the vehicle. Our approach relies on a detector that uses histograms of oriented gradients (HOG), basically following the work by Dalal and Triggs [4]. We selected HOGs as these are gradient-based features and comparably stable under illumination and slight rotation and shape changes. To build a classifier based on the HOG features, we use a standard linear support vector machine. The training is performed using an exiting image database of cars including self-recorded images during several drives through our local city environment. We carry out the detection in a cascade fashion via a sliding window approach.

To compute the relative location of the detected cars, we exploit our stereo setup. Following the work by Konolige [10], we perform block matching along the epipolar lines in the stereo image pairs. Knowing the internal camera parameters, we reconstruct the relative 3D position of each pixel from the disparity image. As our vehicle is equipped with a 2D SICK laser range finder, we combine the stereo depth information with the data from the laser range scanner. This yields the car's positions relative to the camera. To obtain this information in a global reference frame, we integrate the relative poses of cars with the vehicle's pose estimates from a standard graph-based simultaneous localization and mapping (SLAM) system, incorporating also GPS measurements, see [6] for a tutorial. This yields a consistent map and positions of the detected cars in the GPS coordinate frame.

B. Parking Space Modeling

Our planning approach for guiding a vehicle to find a free parking space relies on a graph representation of the environment. Each parking space is modeled through a node and as its occupancy status is generally not known, we use a binary random variable to model it probabilistically.

Throughout this work, we assume that the topology of the environment is given and that the location of parking spaces is known, for example, from open street map. Depending on the level of detail of the map, parking spots may need to be added. In this case, the vehicle only needs to update the occupancy state of the parking spaces it observes.

Assuming that the parking lot is static *over a short period of time*, we can compute its occupancy probability via a static state binary Bayes filter as defined in Eq. (1). We furthermore apply a nearest neighbor data association to assign the detection to parking spaces.

Let $P(x^i)$ be a prior occupancy probability estimate of a parking space i and $z_{1:t}$ be a sequence of observation of this space. According to Moravec and Elfes [12], we can apply the following update rule to calculate $P(x^i \mid z_{1:t})$:

$$P(x^i \mid z_{1:t}) = \left[1 + \frac{1 - P(x^i \mid z_t)}{P(x^i \mid z_t)} \frac{1 - P(x^i \mid z_{1:t-1})}{P(x^i \mid z_{1:t-1})} \frac{P(x^i)}{1 - P(x^i)}\right]^{-1} \quad (1)$$

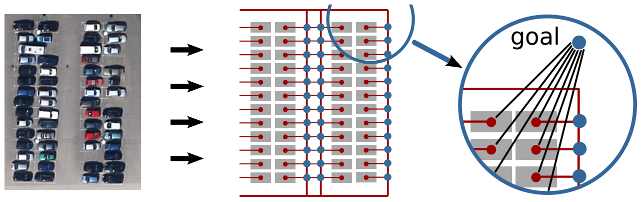


Fig. 2: Left: Example of a parking lot. Right: Graph representing the parking area. Blue nodes are the positions for driving, while red ones denote parking positions. Every parking node is additionally connected to the goal state.

Eq. (1) relies on the observation model $P(x^i \mid z_t)$. Our observation model is similar to a standard raycast model used for laser range finders. All parking spaces in the field of view of the camera are updated up to a given distance. A detection of a car increases the probability that the parking space is occupied while no detection reduces the probability. The data association is done in a nearest neighbor fashion. This allows us to estimate the current occupancy status assuming that the parking lot is static for a short period of time.

C. *Prior for Parking Space Occupancy*

Once we are able to estimate the current occupancy status of a parking lot based on observations as described above starting with $P(x^i) = 0.5$, we can easily learn a parking space specific prior.

Given n data recording sessions, we can use for every session the maximum likelihood estimate of the occupancy probability $P(x^i)$ for parking space i and count the number of times the maximum likelihood estimate indicated an occupied ($N^i_{occupied}$) or free (N^i_{free}) space. This directly yields an occupancy prior for each parking space as

$$P_{prior}(x^i) = \frac{N^i_{occupied}}{N^i_{occupied} + N^i_{free}}. \tag{2}$$

When searching for a free parking space and having not observed the parking space, this term $P_{prior}(x^i)$ provides us with the prior probability that we will experience the parking space with index i as occupied.

IV. Route Planning for Effective Parking

What is a good or even an optimal path to find a free parking spot near to a desired target location? The exact cost function may depend on the application at hand. In this work, our objective function is the time needed to find a parking space plus the time needed to walk to the target location. As the occupancy status of the parking space is not known beforehand, our approach seeks at *minimizing the expected time that is needed to park the car and to walk to the target location.*

A. *Graph-Based Environment Model*

We represent the environment as a graph $G = (V, E)$. Each vertex in $v \in V$ represents a point in the plane ($\mathbb{R}^2$) as a possible location of the car. We distinguish three types of vertices: those that model a parking space (V_p), those that are used to model the possible locations and intersections along the path V_m, and (typically one) goal vertex V_g so that $V = V_p \cup V_m \cup V_g$. Each edge $e \in E$ is a function $V \to V$, that corresponds to the action of moving from one state to another. A graph example for a parking lot is shown in Figure 2. The figure also shows the distinct sets of vertices. The blue vertices are used to model the possible path in the parking lot, while the red ones are parking spaces. Thus, only the red vertices have a connection to the goal state (not shown in middle image).

B. *MDP Definition*

Inspired by the approach of Tipaldi and Arras [20], who present a model for spatio-temporal patterns of human activities that enable robots to blend themselves into the workflows and daily routines of people, we formulate our parking problem as a Markov decision process (MDP). MDPs provide a way to maximize the expected reward given the current knowledge of the world. Following Bellman [2], an MDP is defined by the initial state s_{start}, the transition function $T(s \mid s', a)$ and the reward function $R(s, s', a)$. Here, $T(s \mid s', a)$ denotes the probability of reaching state s if action a is carried out in state s'. The transitions in the model are assumed to be Markovian, i.e., the probability of reaching s from s' depends only on s' and not on the history of previous states.

Actions: In our parking application, we require only a small set A of possible actions. We use four movement actions that correspond to moves in the four cardinal directions $A_{\text{move}} = \{\uparrow, \downarrow, \leftarrow, \rightarrow\}$ plus one parking action a_{park}. Thus, the set A is given by

$$A = A_{\text{move}} \cup a_{\text{park}}. \tag{3}$$

Transition Function: For four actions from A_{move}, we consider the probability of moving from any state to a neighboring one— given a neighbor exists in the corresponding cardinal direction—to be equal to 1, except for the action of moving into a parking space. Here, the probability of being able to move to this state is equal to the inverse occupancy probability of the parking space

$$\begin{aligned} \forall s \in (V_m \cup V_g), \forall a \in A \,&:\, T(s \mid s', a) = 1 \\ \forall s \in V_p, \forall a \in A_{\text{move}} \,&:\, T(s \mid s', a) = 1 - P(s), \end{aligned}$$

where $P(s)$ is the probability of the parking space s to be occupied.

Rewards: We define the reward function such that the MDP seeks to minimize the expected time that the user needs to reach the target location.

Let route S be an arbitrary path to the target location (goal state). This route is effectively described as a sequence of moves that guide the user to the goal. Given our transition model, every move of the agent can be described by two states $s_1, s_2 \in V$ together with an action $a \in A$ that brings the agent from state s_1 to state s_2 with $T(s_2 \mid s_1, a) > 0$. The agent starts in the initial state s_{start} and travels to the goal state s_{goal}. Thus, S is a sequence:

$$S = \big\langle (s_i, s_{i+1}, a) \\ \mid (s_i, s_{i+1}) \in E, \exists a : T(s_{i+1} \mid s_i, a) > 0 \big\rangle_{i=\text{start}}^{\text{goal}}$$

This yields the expected time for the agent to traverse the sequence of states of

$$E(t_S) = \sum_{(s_i, s_{i+1}, a) \in S} T(s_{i+1} \mid s_i, a)\, t_{s_i, s_{i+1}}, \tag{4}$$

where $T(s_{i+1} \mid s_i, a)$ is the probability of moving from state s_i to s_{i+1}, carrying out action a and $t_{s_i, s_{i+1}}$ is the time it takes the agent to travel between the defined states. We seek to find the route S^* that minimizes the expected time defined in Eq. (4):

$$S^* = \underset{S}{\operatorname{argmin}}\, E(t_S) \tag{5}$$

Instead of minimizing the expected time, we can equivalently maximize the negative expected time. Thus, we can define our rewards function $R(s, s', a)$ for reaching state s from state s' carrying out action a as the negative time that the agent needs for driving from state s' to state s. The time is the distance between these states divided by the speed of the agent

$$R(s, s', a) = -t_{\text{drive}}^{s', s} \text{ with } s \neq s'. \tag{6}$$

In addition to that, we also penalize staying in one place by assuming that this takes a constant amount of time t_{wait}. In our system the agent stays in the same state only if he fails to carry out an action, e.g., a_{park} in case of an occupied parking space. This yields

$$R(s, s, a) = -t_{\text{wait}}. \tag{7}$$

After a successful parking action, i.e., being in a state $s \in V_p$, the agent reaches the goal state with probability 1. The reward for reaching the goal state has two components. First, as Eq. (6), we consider the time to *walk* to the target location from the parking space. Second, the agent receives a positive reward $r_{\max}$ for reaching the goal state so that

$$R(s, s_{\text{goal}}, a) = r_{\max} - t_{\text{walk}}^{s, s_{\text{goal}}} \text{ with } s \in V_p. \tag{8}$$

In our system, we define $r_{\max}$ as the walking time from the parking space that is the furthest from the target location, i.e.,

$$r_{\max} = \max_{s \in V} t_{\text{walk}}^{s, s_{\text{goal}}}. \tag{9}$$

This reward function minimizes the expected time needed to find an unoccupied parking space, park the car, and walk to the target location.

C. Solution to the MDP

The combination of transition and reward function guarantees that the agent seeks to minimize driving and walking times, weighted by the probability of the parking space being not occupied.

The key building block of an MDP is the utility function, which is a measure of how good the state is in the long run

$$U^\pi(s) = E\left[\sum_{t=0}^{\infty} R(s_t, s_{t-1}, a) \,\middle|\, \pi, s_0 = s\right]. \tag{10}$$

Given the utility function, we define the policy as an expected sum of the rewards obtained, where the expectation is taken over all possible state sequences that may occur, given that the policy is executed. This yields the optimal policy

$$\pi^* = \underset{\pi}{\operatorname{argmax}} \left[U^\pi(s) \mid \pi, s \in \pi\right]. \tag{11}$$

This is equivalent to solving Eq. (5) resulting in π^* being an optimal decision in the means of minimizing the expected time that the agent needs to find a parking space and walk to the target location. However, as time approaches infinity, the sum in Eq. (10) also tends towards infinitely large values. To avoid this, we make use of discounted rewards

$$U^\pi(s) = E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, s_{t-1}, a) \mid \pi, s_0 = s\right], \tag{12}$$

where $\gamma \in [0, 1)$ is the so-called discount factor.

We solve the MDP using Policy Iteration, see [16] for a good overview. Compared to Value Iteration, Policy Iteration terminates as soon as there is no change to the policy without requiring to compute the exact utility values for all states. In the policy improvement step, we select the optimal action using the maximum expected utility principle in each state, i.e., choose the action that maximizes the expected utility of the subsequent state. The optimal policy $\pi^*(s)$ can therefore be seen as the one that maximizes the expected utility if followed: $\pi^*(s) = \operatorname{argmax}_a \sum_{s'} T(s \mid s', a)\, U(s')$.

This solution to the MDP guarantees to find a path that maximizes the expected discounted rewards. Under the assumption that the discount factor is close to 1, this path is also the one that minimizes the expected time for searching for the parking space and walking to the target location for a given set of parking space occupancy probabilities.

D. Changing Environments and Replanning

The agent carries out an optimal strategy to the point when it decides to park. Now, consider the situation in which the optimal parking space is observed to be occupied. From the point of view of the MDP, the optimal decision would be to try over and over again to park—this is a clearly suboptimal behavior *given that new knowledge is available* through observations

To take observations into account but avoid a POMDP formulation, we continuously update the state of the parking spaces using a Bayes filter and solve the MDP given the updated occupancy probabilities. As result of the update, the occupancy probability of the selected parking space in the previous MDP solution is increased and thus a new policy which guides the vehicle to a different location.

For all parking spaces that the vehicle observes, its status is updated given Eq. (1). This makes the assumption that the

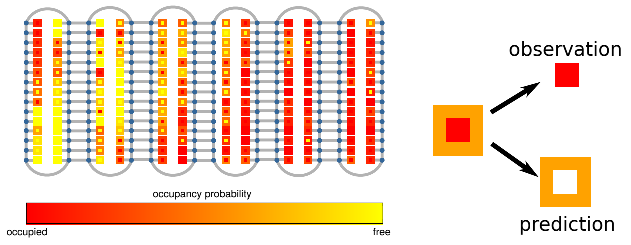


Fig. 3: Occupancy status of the parking space: each inner square denotes current occupancy state using the recursive update, while each outer denotes the prior information from the training phase.

state of the parking lot does not change while the vehicle is observing the scene. In order to consider the fact that cars leave the parking lot and that new cars arrive, we also update the probabilities of the unobserved parking areas. This is performed in the following way.

Let k be the time when the parking space has been observed the last time. We estimate the state of the parking space at time $t > k$ based on the estimated state at time k, i.e., $p(x_k^i \mid z_{1:k})$ and the probability that new cars enter and leave parking spaces.

We model the change in the occupancy variables, i.e., the fact that cars leave or park, after the last observation of that parking space as a Poisson distributed random variable. As a result of that, we can model the probability for change through the CDF of an exponential distribution. This expresses the probability for a change at t_{change} with $k < t_{change} \leq t$

$$P_\lambda(\Delta t) = P(t - t_{change} \leq \Delta t) \quad = \quad 1 - e^{-\lambda \Delta t}, \tag{13}$$

where λ is the frequency at which an occupancy change of a parking space takes place and $\Delta t = t - k$.

Based on Eq. (13), we define a simplified model for non-static parking space occupancy, which is a weighted sum with Eq. (13) being the weight, of the last known belief and the parking space-specific prior $P_{prior}(x^i)$ that models the average occupancy probability for that parking space:

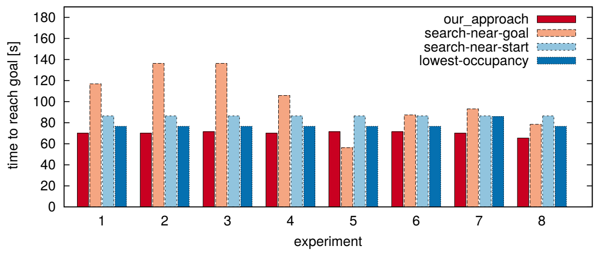
$$p(x_t^i \mid x_k^i, z_{1:k}) \approx$$
$$(1 - P_\lambda(\Delta t))\, p(x_k^i \mid z_{1:k}) + P_\lambda(\Delta t)\, P_{prior}(x^i) \tag{14}$$

Thus, at every time step, the current occupancy probabilities are updated based on Eq. (1) given an observation and using Eq. (14) if the last observation of a parking space is Δt steps old. An example is shown in Figure 3. After updating all parking spaces, a new policy is computed and executed.

V. Experiments

The experiments are designed to illustrate the performance of our approach to autonomous parking and to compare the performance of our method to three manually designed strategies. It is also worth mentioning that the average execution time for the policy update was around 10 ms on a regular computer for all experiments presented in this section.

All our experiments have been conducted on real world data. We recorded the parking lot of our campus over 21 days



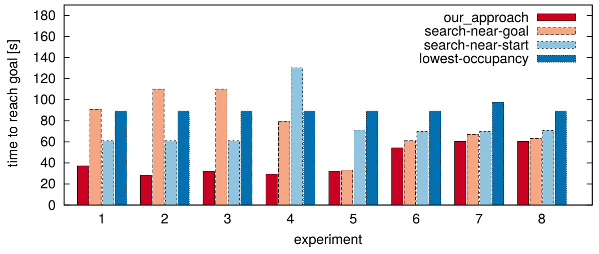


Fig. 4: Comparison of the performance of our approach to three heuristic strategies “search-near-goal”, “lowest-occupancy”, and “search-near-start” in 16 different experiments based on real world occupancy data. As can be seen, our method, which minimizes the expected time to reach the target location, outperforms in general the heuristic solutions—although a heuristic can be better in a specific instance of the parking lot configuration.

using a Bumblebee stereo camera and a SICK laser range finder on our vehicle to compute the prior probabilities for the occupancy. The parking lot has 180 parking spaces and the *outer borders* of the parking spaces in Figure 3 illustrate the results as observed during this period of 21 days. We then executed multiple runs in different settings in which our approach had to compute a path for finding a parking space. The occupancy status of the parking lot according to the observations can be seen from the *inner squares* at the parking space locations.

The first experiment is designed to compare the performance of our approach with respect to three alternative parking strategies. According to our modeling assumptions, our MDP approach computes the optimal path using policy iteration, i.e., the path that minimizes the overall time to find a parking space and to reach the final destination. In order to quantify the gain in time of our solution, we compare it to three heuristics.

The first heuristic called “search-near-goal” drives to the parking space that is closest to the target location and then searches for the next free parking spot in a random walk fashion. The second heuristic called “lowest-occupancy” drives to the parking space with the lowest prior probability of being occupied and then searches for the next free parking spot in a random walk fashion. The third heuristic “search-near-start” looks for a free parking space near the current location. Figure 4 depicts the performance of the three heuristics as well as the one of our approach. As can be seen, our approach outperforms the other strategies in the sense that it guides the user faster to its target location. We also conducted a paired-sample t-test with a 95% confidence level. The test shows that our approach produces trajectories that lead to a significantly shorter time to reach the target location.

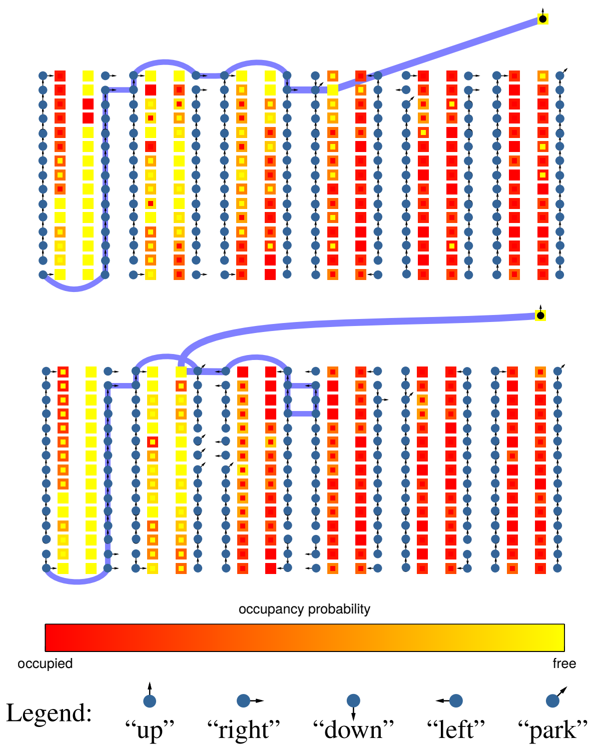


Fig. 5: Example of two full trajectories from the bottom left corner of the parking lot to the goal on the upper right. Both depicted situations share the same initial occupancy prior (outer squares) but differ in the actual occupancy status (inner squares). The upper image shows a situation in which the initial policy of the agent reaches a free parking space making him able to park there, while the bottom image shows multiple re-planning steps the agent takes when the initial optimal parking space is occupied. Eventually, the agent returned to a previously observed free parking space.

The second experiment is designed to illustrate how our approach seeks to find parking spaces. Figure 5 shows two example runs of our method driving to a free parking space and walking to a target location on the top right.

Although the focus of the paper is on the parking space and MDP modeling for effective parking, our approach relies on the ability to detect cars. Using the visual approach described in Sec. III, we achieve a car detection rate of approx. 95%. By additionally using the recursive Bayes filter to integrate the detections over time, we achieve a sufficient performance for building the describes system for effectively finding parking spaces, minimizing the expected time to park and reach the target destination.

VI. Conclusion

Quickly finding a free parking spot close to a given goal location is a desired capability but often hard to achieve, especially in large cities. This holds for human drivers as well as for autonomous cars alike. We investigated the problem of modeling and exploiting the occupancy information of individual parking spaces during route planning. We proposed an MDP-based planner that considers the occupancy probabilities of parking spaces to find the path that minimizes the expected total time spent for searching for an unoccupied parking space as well as for walking from the found parking location to the goal location. We evaluated our system based on real world data gathered with a mobile vehicle over several days in a real parking lot. We compared our approach to three other parking strategies and showed that our method performs significantly better according to a t-test.

References

[1] P. Abbeel, D. Dolgov, A.Y. Ng, and S. Thrun. Apprenticeship learning for motion planning, with application to parking lot navigation. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, Nice, France, 2008.

[2] R. Bellman. A markovian decision process. *Indiana Univ. Math. J.*, 6:679–684, 1957.

[3] V. Coric and M. Gruteser. Crowdsensing maps of on-street parking spaces. In *IEEE International Conference on Distributed Computing in Sensor Systems*, pages 115–122, Washington, DC, USA, 2013. IEEE Computer Society.

[4] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 886–893, 2005.

[5] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel. Practical search techniques in path planning for autonomous driving. In *Proc. of the Int. Symp. on Search Techniques in Artificial Intelligence and Robotics (STAIR-08)*, Chicago, USA, 2008.

[6] G. Grisetti, R. Kümmerle, C. Stachniss, and W. Burgard. A tutorial on graph-based SLAM. *IEEE Trans. Intell. Transport. Syst.*, 2, 2010.

[7] C. C. Huang and S. J. Wang. A hierarchical bayesian generation framework for vacant parking space detection. *IEEE Trans. Circuits Syst. Video Technol.*, 20(12):1770–1785, Dec 2010.

[8] H. Ichihashi, T. Katada, M. Fujiyoshi, A. Notsu, and K. Honda. Improvement in the performance of camera based vehicle detector for parking lot. In *IEEE Trans. on Fuzzy Systems*, pages 1–7, 2010.

[9] J.Z. Kolter, C. Plagemann, D.T. Jackson, A.Y. Ng, and S. Thrun. A probabilistic approach to mixed open-loop and closed-loop control, with application to extreme autonomous driving. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, pages 839–845, 2010.

[10] K. Konolige. Small vision systems: Hardware and implementation. In Yoshiaki Shirai and Shigeo Hirose, editors, *Robotics Research*, pages 203–212. Springer London, 1998.

[11] S. Mathur, T. Jin, N. Kasturirangan, J. Chandrasekaran, W. Xue, M. Gruteser, and W. Trappe. Parknet: drive-by sensing of road-side parking statistics. In *Int. Conf. on Mobile Systems, Applications, and Services*, pages 123–136, 2010.

[12] H. P. Moravec and A. Elfes. High resolution maps from wide angle sonar. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, volume 2, pages 116–121, Mar 1985.

[13] G. Pierce and D. Shoup. Getting the prices right. *Journal of the American Planning Association*, 79(1):67–81, 2013.

[14] G. Pierce and D. Shoup. Sfpark: Pricing parking by demand. *ACCESS Magazine*, 2013.

[15] J. P. Rodrigue, C. Comtois, and B. Slack. *The geography of transport systems*. Routledge, 2013.

[16] S. J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Education, 2 edition, 2003.

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

[18] J. K. Suhr and H. G. Jung. Fully-automatic recognition of various parking slot markings in around view monitor (avm) image sequences. In *IEEE Trans. Intell. Transport. Syst.*, pages 1294–1299, Sept 2012.

[19] J. K. Suhr, H. G. Jung, K. Bae, and J. Kim. Automatic free parking space detection by using motion stereo-based 3d reconstruction. *Machine Vision and Applications*, 21(2):163–176, 2010.

[20] G.D. Tipaldi and K. O. Arras. I want my coffee hot! learning to find people under spatio-temporal constraints. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, Shanghai, China, 2011.

[21] N. True. Vacant parking space detection in static images. University of California, San Diego, 2007.

[22] M. Tschentscher and M. Neuhausen. Video-based parking space detection. *Int. J. of Innovation, Management and Technology*, 2012.

[23] Qi Wu and Yi Zhang. Parking lots space detection. *Machine Learning, Fall*, 2006.