Parking Spot Estimation and Mapping Method for Mobile Robots

Thomas Westfechtel, Kazunori Ohno, Naoki Mizuno, Ryunosuke Hamada, Shotaro Kojima, and Satoshi Tadokoro

Abstract—Self-driving vehicles rely on detailed semantic maps of the environment for operating. In this letter, we propose a method to autonomously generate such a semantic map enriched with knowledge of parking spot locations. Our method detects and uses parked vehicles in the surroundings to estimate parking lot topology and infer vacant parking spots via a graph-based approach. We show that our method works for parking lot structures in different environments, such as structured parking lots, unstructured/unmarked parking lots, and typical suburban environments. Using the proposed graph-based approach to infer the parking lot structure, we can extend the estimated parking spots by 57%, averaged over six different areas with ten trials each. We also show that the accuracy of our algorithm increases when combining multiple trials over multiple days. With ten trials combined, we managed to estimate the whole parking lot structure and detected all parking spots in four out of the six evaluated areas.

Index Terms—AI-based methods, big data in robotics and automation, intelligent transportation systems, mapping, semantic scene understanding.

I. INTRODUCTION

Over the next few years, self-driving vehicles will become an increasingly common sight on public roads. In order for them to operate in complex environments, detailed semantic maps of the environment are necessary. Depending on the scope and details of the semantic information, the generation of a map can be time consuming and may require the deductive capabilities of humans. At the same time, autonomous vehicles will share the same environment with human-operated vehicles. In this work, we wanted to draw upon the knowledge of human drivers and extract semantic knowledge about parking spots by observing their behavior, especially where the drivers park their vehicles. Finding a suitable parking spot is an essential task for self-driving cars. To put this into perspective, modern advanced driver-assistance systems (ADAS) are capable of autonomously executing a parking maneuver, but the parking spot has to be localized by the human operator first. This task can be tedious and time-consuming. According to a study by Arnott and Inci [1], nearly 30% of traffic in cities is caused by car drivers that are searching for a parking spot.

In this letter, we propose a method to autonomously generate a map of the environment and enrich it with information about parking spot locations. Our method uses parked vehicles in the surrounding area to estimate the parking lot topology and infer vacant parking spots via a graph-based approach. We employed an autonomous electrical vehicle (AEV) representatively for future self-driving vehicles to acquire data. The AEV was equipped with a multitude of sensors including a Velodyne HDL-32E LIDAR (see Fig. 2). The LIDAR data was used to generate a map of the environment using a simultaneous localization and mapping (SLAM) algorithm and simultaneously was used to detect and localize vehicles using a convolutional neural network (CNN). The results were fused and the topology of parking lots were estimated and vacant parking spots were inferred in a graph-based approach.

This work is licensed under a Creative Commons Attribution 3.0 License. For more information, see http://creativecommons.org/licenses/by/3.0/
In addition, we show that by observing the same area over multiple days, our method can not only infer most, if not all, of the parking lot structure, but the results can also be used to evaluate the rate of occupancy of parking spots on different weekdays and at different times of day. This can, in turn, help alleviate the task of finding a parking spot not just for self-driving cars but also for human drivers.

The task of parking spot detection is not trivial. While parking spots are often outlined by road markings, this is not always the case. For example, in one of our experimental areas (see Fig. 8a) some of the parking spots were only marked by a stretched thin string while other parking spots had no markings at all. In addition, during some of our experimental trials, the ground was covered with a few centimeters of snow, blocking any visual features. Another challenge arises from the combination of different core technologies, each of them also introducing uncertainties. For example, we used a CNN that was trained on the KITTI dataset for a Velodyne HDL-64E sensor. However, in our experiments we employed a Velodyne HDL-32E sensor, which has a vertical field of view that is 1.5 times higher than that of the Velodyne HDL-64E sensor, but only approximately one third of its vertical resolution. This impacts the average precision of our system. To make these challenges even harder, we did not employ any prior knowledge and only used the data acquired by the vehicle’s LIDAR (and GPS data for pre-registering different trials on different days).

The main contributions of this work are the following:

- A method that adds semantic knowledge to the environment for the specific case of parking spots.
- The combination of different core technologies to detect parking spots by observing other traffic participants, specifically parked vehicles.
- A graph-based method that estimates the topology of parking lots and infers vacant parking spots.
- A system that simultaneously generates a map of the environment and enriches it with parking spot locations.
- A framework that can be used to evaluate the occupancy of parking spots at different times, which is the basis for predicting the availability of open parking spots.

II. RELATED WORKS

Semantic knowledge acquisition from mobile sensor data is a hot and important research topic in data engineering, artificial intelligence and robotics. Yang et al. [2] presented a method for building 3D semantic maps from stereo image streams. In their work, the researchers categorized each point of a 3D map into one out of ten groups titled, for example, building, road, vegetation, car, sidewalk, or fence. The overall goal was to build geometric maps for mobile robots to enable robots to not only avoid obstacles, but also to recognize objects for high-level tasks. Kochanov et al. [3] followed a similar approach, but placed emphasis on temporal tracking and updating dynamic (moving) objects in a temporally and spatially consistent semantic 3D map of the surroundings. Their main motivation was that autonomous vehicle need to operate in a dynamic environment. The vehicle therefore needs to understand these dynamics to a large degree. Another research in this field was carried out by Sünderhauf et al. [4]. The researchers focused on building environmental maps that combined semantic meaning in the form of object classifications, such as “monitor” or “keyboard”, with the point- or mesh-based representations of the objects and the environment. They placed focus on the fact that robots need to have knowledge about their environment in form of semantic meaning but at the same time also need to interact with the objects, and therefore require an accurate geometrical representation of them. Another interesting work was presented by Liu et al. [5]. The researchers extracted generalized motion patterns of a typical traffic environment. In contrast with other presented works, the extracted knowledge was used in a subsequent step to detect and extract regions of interest, such as lane merges, lane diverges, and intersections. Therefore, they employed learned semantic knowledge to further infer high-level knowledge.

From these related works, the scope and difficulty of semantic mapping and scene understanding becomes apparent. For mobile robots to reliably act and interact in a human environment, high-level knowledge about the environment itself, its context, and the dynamics of objects are indispensable. In this work, we wanted to tackle this problem for the specific case of parking spot detection. We gathered knowledge about the environment in the form of potential parking spots by observing parked vehicles of other traffic participants. We put this knowledge in context via a graph-based approach to estimate the underlying parking lot structure, thus creating a map of the environment enriched with parking spot knowledge.

In the next part, we present related works for the specific case of parking spot detection. Most researchers have made use of the white line markings of parking spots for this task. Seo and Urmson [6] used aerial images to detect the structure of a parking lot and its parking spots. In the first stage, single parking spots were detected using the line markings. These detections were merged into parking blocks. Using the spatial relations between the parking blocks, additional hypotheses for parking spots were generated, which were then verified or discarded using a machine learning technique. While this is an interesting approach, we are more interested in on-vehicle solutions. An algorithm for parking spot detection for vehicles was presented by Houben et al. [7]. In their work, the researchers performed a coarse detection of parking spot line markings via a local symmetry measure for a fisheye camera. The line detections were used to estimate the parking lot row patterns and to predict possible distant parking spots. Hamada et al. [8] employed...
Fig. 3. Flowchart of our parking spot detection method.

a similar method using a surround-view constructed from four RGB cameras. The researchers use edge detection and line extraction in combination with a probabilistic Hough transform to detect lane markings. The detected markings were then compared to common parking spot markings for detection. In all of these works, the line markings of parking spots were used. However, in real-world scenarios, these markings are not always available. In some cases, the markings are replaced with other indicators, such as a stretched string, or are completely missing. In such cases, these algorithms would not be able to detect the parking spots.

The research that is probably most closely related to our work was presented by Zhou et al. [9]. In their research, the authors used AdaBoost to extract the line segments of car bumpers from scans acquired using a 2D LIDAR installed on a mobile robot. These detections were used as vehicle hypotheses to create a topological structure of the parking lot and detect open parking spots. However, in their work, the authors assumed that their robot was located within a parking lot. It is unclear whether their algorithm would work in a normal street environment. Furthermore, in our work, we focused not only on the detection of parking spots, but also on creating a semantic map using these pieces of information. Additionally, we revisited the same location several times to improve the estimation results and evaluate the occupancy rate for detected parking spots.

III. METHODOLOGY

For our parking spot detection and prediction method, different core technologies were combined. A coarse flowchart of our processing stream can be seen in Fig. 3. The algorithm can be divided into three different modules.

The first module generates a 3D map of the environment. The map is transformed into a 2D grid representation to compress the data size. The maps constructed from several trials over multiple days are merged into a single map in order to remove dynamic obstacles. The second module detects vehicles parked by other traffic participants in the surroundings and projects them onto the acquired map. Finally, the third module estimates the parking lot topology by employing a graph-based structure that combines the vehicle detections with the 2D grid map. Using the inferred topology, the method predicts vacant parking spots. In the following sections, we explain each module in more detail.

A. First Module: Environment Map Generation

To generate a 3D map of the environment, we employed the LOAM [10] algorithm for SLAM. LIDAR points measuring the host vehicle were removed in a preprocessing step to increase the robustness of the SLAM algorithm. To decrease the size of the pointcloud and to have a more balanced distribution of points, we downsampled the pointcloud using an octree to one point per 1 cm$^3$. The results can be seen in Fig. 4(a).

In order to extract the drivable area, we employed a region growing algorithm on the pointcloud to detect the ground surface. For this region growing algorithm, we defined three termination conditions.

- The point normal is not facing within $\pm 20^\circ$ in the direction of the z-axis, i.e., the vertical axis.
- The difference between the point normal of two neighboring points is above $5^\circ$.
- The difference in surface level between two neighboring points is above 10 cm.

The difference in surface level $d_s$ is calculated as

$$d_s = (p_i - p_j) \cdot \mathbf{n}_i$$

where $p_i$ and $p_j$ are the coordinates of the neighboring points and $\mathbf{n}_i$ represents the point normal of point $i$. The region consisting of the most points was assumed to depict the ground surface. All other points were assigned to be part of the object cloud. The results of this ground surface extraction process can be seen in Fig. 4(b).

In order to generate a 2D occupancy grid map from the pointcloud, a map for the area under investigation is first generated, in which each grid cell is marked as unknown area. We iterated
occupancy grid maps, free grid cells have priority over occupied grid cells, which themselves have a priority over unknown grid cells. While this simplistic priority ranking proved sufficient for the current scope of our experiments, we plan to employ a more sophisticated probabilistic method for the fusing process in the future.

B. Second Module: Vehicle Detection Module

For the detection of vehicles, we employed the CNN-based DoBEM method [12], which uses elevation maps generated from range data as input. We chose this method because it is able to simultaneously detect and localize vehicles in the horizontal plane. It should be noted, however, that the CNN was trained using the KITTI dataset, which uses data from a Velodyne HDL-64E sensor, and that our electric vehicle was equipped with a Velodyne HDL-32E sensor. As previously stated, while the Velodyne HDL-32E sensor has a vertical field of view that is 1.5 larger, its vertical resolution is only about one third of that of the Velodyne HDL-64E sensor. This lowers the detection rate, especially for vehicles at a larger distance from the host vehicle. In order to overcome this challenge, we combined the last $n$ registered scans to generate a single elevation map. Additionally, in the KITTI dataset, only the forward-facing part of the pointcloud (the direction to which the host vehicle is heading) was used for training and evaluation. To use the full $360^\circ$ roundview, we flipped the backward-facing part (w.r.t the Velodyne LIDAR) of the pointcloud and processed it in the same way as the forward-facing part. For the forward-facing elevation map, we used the host vehicle position corresponding to the earliest of the $n$ scans, while, for the backward-facing part, we used the latest. As we assume that the vehicle is driving in forward direction, this guarantees that most 3D points will be incorporated in the elevation map. The aggregated detection of a single trial can be seen for $n = 1$, $n = 10$ and $n = 20$ on the left side of Fig. 5. Vehicles further away from the host vehicle’s trajectory (highlighted by an orange box) were not detected for $n = 1$, while they were mostly detected for $n = 20$. On the other hand, the amount of false positives also increased with $n$. In our experiments, we found that a value of $n = 10$ results in the best trade-off between false positives and false negatives.

Because we employed the CNN with different settings than those it was trained for (different LIDAR sensor and agglomeration of multiple scans), false positives were not uncommon. To remove false detections and cluster the multiple detections of a single car, we employed the density based spatial clustering of applications with noise (DBSCAN) algorithm [13]. As a positive side effect, vehicles in motion are also removed, as they are only observed once in a given location. Each cluster is merged to an estimated position $\hat{M}(x, y)$, which is calculated by averaging the center points of the single detections $\hat{M}_i(x, y)$. For estimating the heading direction of the vehicle, we assumed that the results of the CNN exhibited a heading direction estimation error around the true heading $\theta$, and that this error had a Gaussian shape $\mathcal{N}(\theta, \sigma^2)$. For simplicity, we assumed a variance of $\sigma = 10^\circ$. We estimated the heading $\hat{\theta}$ by maximizing the sum
of the Gaussian probability density function, expressed as \( f() \), for all observed angles \( \theta_i \):

\[
\hat{\theta} = \arg\max_{\theta} \sum_{i=1}^{n} f(\theta|\theta_i, \sigma^2)
\]

The equation was solved numerically and \( \hat{\theta} \) was used as the estimated vehicle heading direction (see Fig. 6b). This definition reduces the weight of outliers on the results compared with an empirical mean. The resulting merged detections can be seen for \( n = 1 \), \( n = 10 \) and \( n = 20 \) on the right side of Fig. 5. It can be seen that even for \( n = 20 \) all false positives were removed; however, two true positives were also removed (see orange box).

**C. Third Module: Parking Lot Graph**

With the detected parked vehicles, we spanned a graph connecting the vehicles in order to estimate the topology of the parking lot. The center point of each car represents a graph node, while the connection between two cars represents a graph connection. In this work, four different types of connections were differentiated.

- **L<sub>SBS</sub>:** Two cars Side By Side, occupying neighboring parking spots.
- **L<sub>SVS</sub>:** Two cars Side, with a Vacant spot in between, to Side.
- **L<sub>NBN</sub>:** Two cars Nose By Nose, occupying parking spots facing each other.
- **L<sub>NVN</sub>:** Two cars Nose, with a Vacant area in between, to Nose.

We did not distinguish between the front and the back of each car; we used the term "nose" for both of them. Furthermore, for these connections, we define side connections to be within 30° to the width alignment of the vehicle and nose connections to be within 30° to the length alignment of the vehicle, respectively.

To detect the connections, we used the recommended parking spot sizes for Japan, provided by the Japanese Ministry of Land, Infrastructure, Transport and Tourism (MPLIT). We used the recommended width of 2.5 m, with a margin of 1.5 m to account for deviation from the recommended size and to take into account the fact that cars are not always parked in the center of the parking spot. A **L<sub>SVS</sub>** type connection spans an additional distance of the recommended 2.5 m parking spot width.

For the **L<sub>NBN</sub>** and **L<sub>NVN</sub>** connections, we used the recommended length of a parking spot of 6 m accordingly. We employed a slightly larger margin of 2 m to account for cars with a smaller length.

We further enforced that neighboring parking spots need to have the same heading direction. To account for sloppily parked vehicles, as well as estimation errors arising from the car detection method, a heading difference of 20° was accepted.

Examples of the four different graph connections can be seen in Fig. 7a. Nodes connected through **L<sub>SBS</sub>** connections are defined as a parking row. In addition, nodes that are connected via an undefined amount of **L<sub>SBS</sub>** connections are not allowed to have a connection of any other type with each other.

1) **Extension and merging of parking rows:** In the first step of this process, **L<sub>SVS</sub>** connections are analyzed. These connections often depict three successive parking spots, with the middle one being vacant. Therefore, a parking spot is assumed to be in the middle of the two nodes (vehicles). If the car does not overlap with obstacles in the grid map, the prediction is accepted, otherwise it is rejected. In cases in which the prediction is accepted, the **L<sub>SVS</sub>** connection is replaced with two **L<sub>SBS</sub>** connections, connecting the predicted parking spot with its neighbors. This can be seen in Fig. 7b (note the changes from the dark green **L<sub>SVS</sub>** connections to red **L<sub>SBS</sub>** connections).

In a subsequent step, each parking row is extended. The average distance and direction between neighboring parking spots (spots connected by an **L<sub>SBS</sub>** connection) is calculated and, using these values, the graph is extended in both directions. The extension is halted if obstacles or a driving lane is encountered. For driving lanes, an additional 50 cm safety margin is added. The extension is carried out for up to three parking spots to each side. During the extension process, each predicted parking spot is analyzed for **L<sub>SVS</sub>** connections and, if they are found, the respective parking rows are merged (given the absence of obstacles).
In parking lots, each car has to be able to leave its parking spot and the lot at any time. This means that each parking spot has to be connected to a driving lane. We can exploit this relationship to predict the driving lanes of a parking lot. Each parking row is therefore required to be connected to a driving lane running parallel to it. We assumed a driving lane with a width of 2 m parallel to each side of the parking rows. The lane must not overlap with any obstacles or detected vehicles. If driving lanes on both sides of the parking lane satisfy this condition, priority is given to the driving lane that is crossed by \( L_{NVN} \) connections. In such cases, the vacant area between the vehicles is interpreted as a driving lane. In the final step, if the endpoint of a predicted driving lane is within a small distance to the trajectory of the host vehicle, we connect the lane to the trajectory, given that the connection is not obstructed by any obstacle. The predicted driving lane has priority over predicted parking spots, meaning previously predicted parking spots are erased if they overlap with the predicted lane. The results of the lane prediction process can be seen in Fig. 7e.

D. Combining Multiple Runs

For combining multiple trials, we aggregated the clustered vehicle detections of the single trials. The detections were clustered again in a similar fashion to Sec. III-B. This process can lead to contradicting results, e.g. when some vehicles were parked between two parking spots during a trial, ignoring the parking lot layout. To remove ambiguities, we detected overlapping detections on the occupancy grid map after the clustering. From the overlapping clusters, the clusters consisting of car detections are considered to be less likely and are therefore removed.

IV. EXPERIMENTS AND RESULTS

For our experiments, we acquired data from three different locations within Sendai, Japan. We used an electric vehicle, equipped with a Velodyne HDL-32E sensor for data acquisition (see Fig. 2). For each location, we acquired data of the same route on 10 different days. The first two experiment locations are parking areas at the Tohoku University (see Fig. 8a and Fig. 1). While the first location consists of mostly structured parking spots, the second location exhibits mostly unmarked, not strictly structured parking places. The third location is a typical suburban area with a few smaller shops (see Fig. 8b). In the pictures, the detected and inferred parking spots are shown, as well as the predicted driving lanes. The detected parking spots are colored according to their occupancy rate (i.e., how often the spots were occupied during the 10 trials), from a bright orange for a low occupancy rate to red for a high occupancy rate. The limitations of our algorithm are also visible in the same figure just above Area 1. This area also consists of several parking rows. However during our 10 trials, only one of the many parking spots was occupied. Without at least a few parking spots being occupied, we currently cannot infer the structure of that portion of the parking lot.

Our algorithm mostly infers parking spots correctly. However, in some cases, the parking rows are extended too far. As we are currently only using a LIDAR for our algorithm, it can be very difficult to correctly detect the end of a parking row, because it does not necessarily coincide with obstacles or even a change in the structure of the ground. This can be seen on the leftmost parking row of Area 2 in Fig. 1. All three of the inferred parking spots at the upper part and two of the inferred spots at the lower part of the row are incorrect.

In Fig. 9, the results for the parking spot detection process for single trials are compared with the combined trials for Area 5. It can be seen that while for each single trial only part of the
parking lot was detected and inferred, the combination of all trials let us detect the whole structure of the parking lot. Thus, the advantages of combining the data of several trials becomes evident. It is also interesting to see that two of the parking spots were not occupied in any of the trials and, thus, detecting these spots still relied on the parking lot topology estimation.

In the second part of our experiments, we evaluated the correctly detected and correctly inferred parking spots, as well as the falsely inferred parking spots, for the highlighted areas of the locations shown in Fig. 1 and Fig. 8. We evaluated the areas for the 10 accumulated trials as well as for each individual trial by itself. The results are shown in Fig. 10. In four of the six areas the whole structure of the parking lot could be estimated when using the accumulated trials. In Area 2, all but two parking spots were detected. Additionally, for the single trials, we increased the estimated parking spots by 57% using our graph-based approach. In some cases, the estimated parking spots were more than doubled. However, more importantly, in more than 90% of the cases at least one vacant spot was inferred from the parking lot structure, meaning that an autonomous vehicle would have been able to accomplish the task of autonomously finding a parking spot.

It should be noted that, on some occasions, parking spots were falsely inferred. This happens mainly when parking rows are extended beyond their endpoint. One could argue that these predictions are nevertheless interesting, as many human drivers would still use such spots if no designated parking spots were vacant. It is debatable whether this kind of behavior is desired or not.
Furthermore, it is very interesting to see the huge drop in the occupancy of parking spots during the weekend for Areas 1–4. While this is not surprising (as most students and professors rest during the weekend), such information could be used to classify parking lots as industrial (high occupancy during weekdays, low occupancy during weekends), recreational (low occupancy during weekdays, high occupancy during weekends), and residential (stable occupancy throughout the week) parking lots. Furthermore, in Area 4, a significant drop can be observed on one Wednesday. By looking at our acquisition logs, we discovered that it had snowed during the morning of that day.

V. CONCLUSION AND DISCUSSION

In this work, we presented an algorithm for generating semantic maps of the environment enriched with parking spot locations. In the first step, we used a SLAM algorithm to register the single scans of a LIDAR sensor. These scans were used in two ways. First, the scans were used to generate a 3D map of the environment. At the same time, \( n \) registered scans were utilized to generate elevation maps as input for a convolutional neural network to detect vehicles. In a subsequent step, the 3D environment map was processed to extract a 2D occupancy grid map, including the trajectory of the host vehicle. To remove dynamic obstacles, the grid map of several trials over the same area were fused. The vehicles detected by the CNN were clustered and projected onto the grid map. In the last step, we presented a graph-based approach to infer the topology of the parking lot and estimate vacant parking spots.

We demonstrated that our algorithm can infer much of the parking lot structure even from a single trial given that at least a few parking spots are occupied. In many cases, we could double the estimated parking spots with our graph-based approach. On average, over 6 areas with 10 trials each, we could increase the estimated parking spots by 57%. In addition, we showed that combining several trials improves the results, and that in four out of six cases the whole parking lot structure could be estimated with only a few falsely inferred parking spot estimations. We believe that our algorithm is very valuable for the emergence of self-driving cars, because finding and entering a parking spot is an essential task for such cars.

In a subsequent step, we also demonstrated that, using our method, a detailed map showing the occupancy rate for parking spots can be generated. We believe that in a larger setting with multiple vehicles using the same technique and sharing their acquired data a very detailed vacancy probability for each parking spot for different times of day could be created. Using this information, the vacancy of parking spots could be predicted, greatly helping not just self-driving cars, but also human drivers, to find a parking spot.

In our work, we focused on perpendicular parking, which is most common in parking lots and is also the most dominant form of parking in our experiment locations. In general, our algorithm can be extended to also include parallel parking and angled parking. However, this also introduces some ambiguities, e.g., an \( L_{NB} \) connection could be either interpreted as two parking rows facing each other or as two cars of a parallel parking line. Another ambiguity can be seen in Fig. 11.

In our current work, we only used the information gathered by a LIDAR, which has the huge advantage of illumination invariance and can accurately measure the shape of the environment. In future works, we intend to also incorporate features or knowledge extracted from the video stream of surround-view cameras. Furthermore, we are interested in detecting and analyzing the trajectories of other traffic participants, as we believe this holds valuable information about the environment and how humans interact with it.

REFERENCES