

INFORMATION AND COMMUNICATION TECHNOLOGIES
COLLABORATIVE PROJECT

“V-Charge”

Autonomous Valet Parking and Charging for e-Mobility

Grant Agreement Number 269916



V-CHARGE

Deliverable Number	4.4
Deliverable Name	Final integration and evaluation of global and partial planning
WP Number	4
Date of Preparation	October 28, 2015
Date of Submission	October 31, 2015
Editor/Beneficiary	U. Schwesinger / ETH Zurich C. Last, W. Derendarz / VW

Contents

1	Executive Summary	2
2	Integration Status	3
3	Navigation Modules	4
3.1	Global Task Planner (VW)	5
3.1.1	Re-planning of global path	6
3.1.2	Computation time	8
3.2	Online Partial Motion Planning (ETHZ-ASL)	9
3.2.1	Semantic Map Usage	9
3.2.2	Intersection Handling	9
3.2.3	Vehicle Movement Predictions	10
3.2.4	Evaluation	10
3.2.5	Conclusion	14
3.3	Off-Lane/Parking Planner (VW)	15
3.3.1	Overview of the Approach	15
3.3.2	Experimental Evaluation	19
3.3.3	Conclusion and Future work	24
4	3D Multilevel Operation	25
5	Conclusion	28
	References	29
A	Vision-Only Fully Automated Driving in Dynamic Mixed-Traffic Scenarios	30
B	Fast Collision Detection through Bounding Volume Hierarchies in Workspace-time Space for Sampling-based Motion Planners	32

1 Executive Summary

The goal of the V-Charge project is to develop automated automotive navigation functionality within a restricted hybrid structured-unstructured parking lot area. Automated cars will approach, charge, and return from parking zones while safely avoiding other cars and traffic participants. One of the key characteristics of the project is that localization, mapping, and obstacle detection are solely achieved with the help of low-cost vision sensors – supported by sensors already available in many cars, including wheel encoders, IMUs, and ultrasound sensors.

Workpackage 4 “Motion Planning and Vehicle Control” deals with the development of motion planning and vehicle control algorithms suitable for automated driving in the presence of other vehicles and pedestrians:

- Task 4.1: Specification of motion planning and vehicle control concepts (ETHZ (Lead), VW, RBG)
- Task 4.2: Global path planning (ETHZ (Lead), VW, RBG)
- Task 4.3: Control Interface and trajectory following (VW (Lead), ETHZ)
- Task 4.4: Partial motion planning and collision avoidance (ETHZ (Lead), VW, RBG)

Task 4.1 has been thoroughly addressed by deliverable D 4.1, and the approaches taken within the V-Charge consortium to tackle tasks 4.2-4.4 have been presented in deliverables D 4.2 and D 4.3, respectively. In the project’s fourth year, we robustified platooning in the presence of other vehicles and incorporated safe navigation among pedestrians and at intersections. Parking accuracy has been improved even further, enabling precise manoeuvres to perfectly position the vehicle on inductive charging plates.

Milestone 5 does not contain any tasks for workpackage 4. Hence this deliverable reports on the tasks specified in milestone 6. Milestone 6 specifies three tasks to be achieved for workpackage 4 “Motion Planning and Vehicle Control”. They are summarized in [Table 1](#). These tasks address global motion planning utilizing semantic information, vehicle trajectory control with tight error bounds, and partial motion planning in mixed-traffic scenarios – which includes estimating and incorporating other participant’s intentions and semantic information.

The deliverable at hand will address the “final integration and evaluation of global and partial planning”. For this purpose it details progress and efforts undertaken since the last reporting period. Also it contains quantitative and qualitative evaluation data. We will refer to the quality measures specified in milestone 6 in this deliverable.

Task	Description
Task 4.2: Global path planning	Motion planning functional using the two-dimensional (2D) occupancy map from T2.1, semantic information from T2.3, and sampling strategies from T2.4. Reaction to real-time goal attribution including parking and/or charging station maneuvers. Planning in a three-dimensional (3D) environment. Planning time < 5 s, required clearance < 10 cm, maximum velocity 20 km h^{-1} .
Task 4.3: Control Interface and trajectory following	Trajectory following functional with less than 20 cm tracking error in general navigation, less than 5 cm tracking error for maneuvers, management of abrupt change of goal, discontinuities, trajectories indoor+outdoor.
Task 4.4: Partial motion planning and collision avoidance	Local obstacle avoidance using real-time object detection and tracking from T3.3 and T3.4, management of multiple pedestrians. Interaction with navigating and maneuvering cars. Integration of other V-Charge vehicle intentions, integration of semantic information (T2.3) for pre-braking. Max obstacle velocity 10 km h^{-1} . Reaction time < 50 ms.

Table 1: Task specifications for milestone 6 in workpackage 4 “Motion Planning and Vehicle Control”.

2 Integration Status

Within workpackage 4, different navigation modules have been developed. Table 2 lists these modules and their integration status:

Task	Category	Module	Responsible Partner	Integration Status
4.2	Global Path Planning	Mission Executive	VW	Integrated
		Global Planner	VW	Integrated
		Parking/Off-lane Planner	VW	Integrated
4.3	Control Interface and trajectory following	On-lane controller	ETHZ-ASL	Integrated
		Parking controller	VW	Integrated
4.4	Partial motion planning and collision avoidance	Local On-Lane Planner	ETHZ-ASL	Integrated
		Dynamic Object Prediction	ETHZ-ASL	Integrated

Table 2: Overview of the navigation modules developed in workpackage 4.

Global path planning (task 4.2) is tackled via the modules mission executive, global planner, and parking planner. All of them are all fully integrated on the platform. The evaluation of the global planner and parking planner are presented in [Section 3.1.2](#) and in [Section 3.3.2](#), respectively.

Task 4.3 addresses the control interface to the car as well as trajectory following. Two modules contribute to this task. Both are fully integrated on the platform and results are given in [Section 3.2.4](#) as well as in [Section 3.3.2](#).

Local partial motion planning is tackled in task 4.4 with the modules for on-lane planning and dynamic object prediction, likewise all integrated on the platform. An evaluation is presented in [section 3.2.4](#).

3 Navigation Modules

With the submission of deliverable D 4.1, the V-Charge navigation architecture specification has been finalized. However, for the sake of completeness, we shortly review the navigation framework in the following. Figure 1 displays the conceptual layout of the navigation framework, including adjacent modules. The navigation framework includes a behavioral layer that handles priority between a set of specialized planning instances (called task processors). Due to the special setup of the V-Charge project centered around a parking lot, modules operating on the central server side (light blue) are distinguished from those operating on each of the automated vehicles (dark blue).

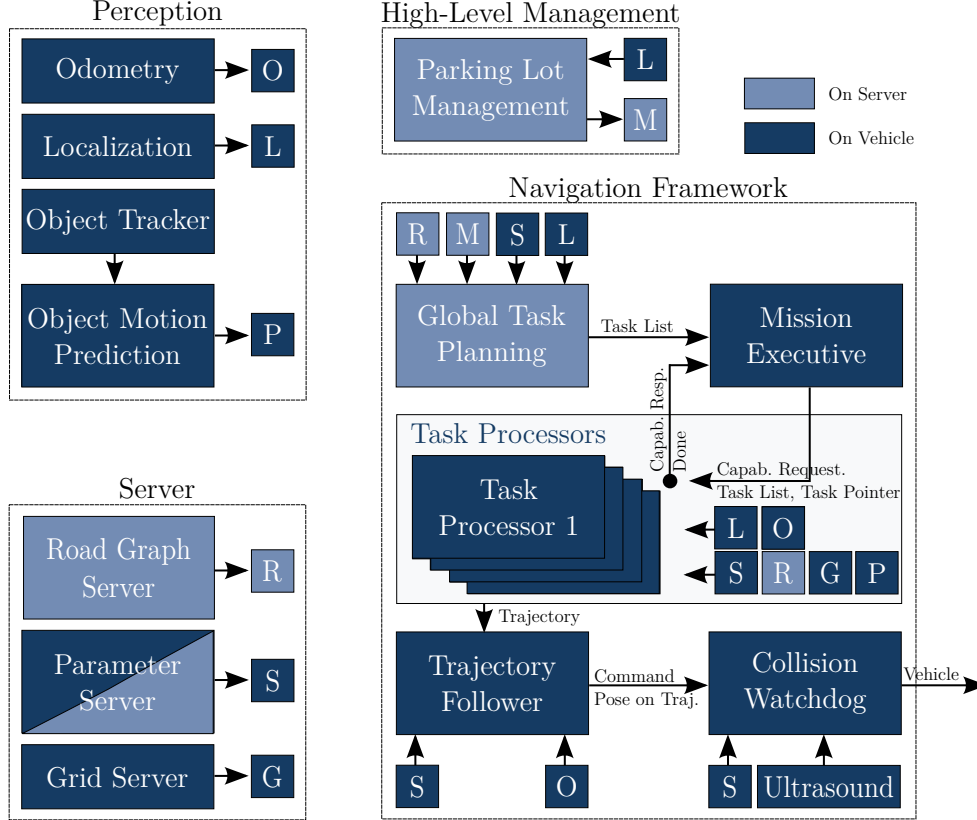


Figure 1: The V-Charge navigation architecture, including adjacent modules.

- The *parking lot manager* is responsible for processing requests from incoming and outgoing vehicles. It assigns free parking spots and charging areas by considering charging needs and expected parking time. It is furthermore responsible for ordering vehicles back to the pick-up zone upon request. The parking lot manager communicates the resulting mission goals to the global task planner on the target vehicle.
- The *global task planner* operates on a regularly updated road graph (obtained from the road graph server). It is responsible for topological route planning on the road-graph. Once a solution to the mission is available in the form of a sequence of topological edge transitions, the global task planner divides it into a feasible sequence of tasks that will be executed by specialized task processors.
- The *mission executive* is responsible for the assignment of the tasks to the individual task processors, management of task processor exceptions and the overall correct execution of events. It contains a task scheduler in the form of a state machine which advances a task pointer once a given task has been completed by the responsible task processor.

- The *task processors* are responsible for one or more task types. In the V-Charge framework, three task handlers are used to cover the full spectrum of situations, namely an online partial motion planner for road-following tasks, a parking planner for parking tasks and a dedicated intersection module.
- The *trajectory follower* serves as an abstraction to the vehicle’s low-level command and control infrastructure. It receives and processes a standardized trajectory description that needs execution from the currently active task processor (road following or parking module).
- The *collision watchdog* ensures a last line of safety based on high-frequency local sensor information.

In the following subsections, we detail final improvements and evaluation results of the global task planner (Section 3.1), the online partial motion planner (Section 3.2), and the parking planner (Section 3.3), respectively.

3.1 Global Task Planner (VW)

Taking into account all static information stored in a road graph, the global task planner is responsible to provide a global reference path that guides the automated vehicle through the parking lot environment. In the previous reporting period, the global task planning module has already been completed. For convenience, we provide a summary of the most important aspects of this module:

- Global task planning operates on request only. It is triggered by the parking lot manager when the user sends parking or pick-up requests from his or her smartphone, when it is time to charge the vehicle, or when the local trajectory planner is unable to further progress along the solution path.
- The global task planner returns a kinematically feasible path from the current vehicle pose to a (set of) feasible goal pose(s), determined by the parking lot manager. The solution path is split into a sequence of tasks which are then passed on to the mission executive.
- The global task planner takes into account the static environment that has been surveyed and processed into a road graph, a graph structure representing the traffic infrastructure, offline. Dynamic objects are neglected in the global planning stage.
- The global planning process is subdivided into two consecutive steps. At first, a simple topological search of the road graph yields a set of edges that guide the vehicle from the start pose to the goal pose. In a second step, the selected edges are smoothed in order to obtain a drivable path (see Figure 2).

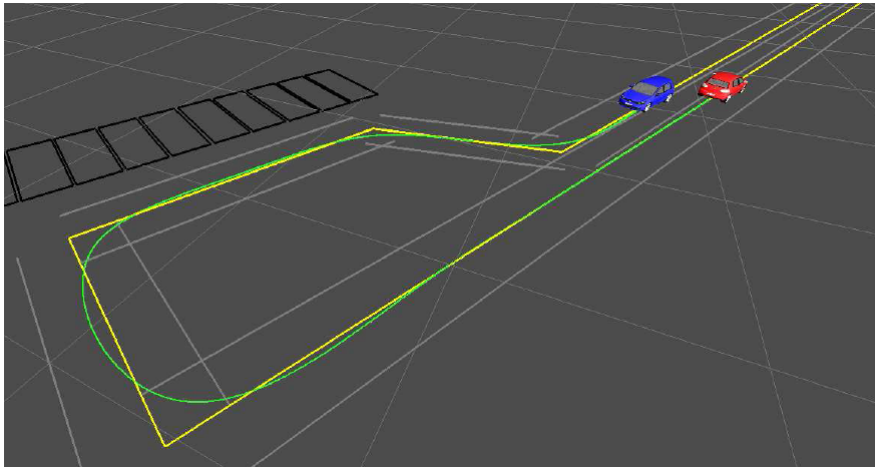


Figure 2: Example of a manually constructed road graph (yellow) together with the smoothed global path (green) from the current vehicle pose (blue) to the target pose (red).

In the 4th period, the only minor adaptations of the global task planning module included:

- An additional task type has been added to handle intersections properly. Such an *intersection task* is passed to a specific *intersection handler* which is responsible for monitoring the intersection for other traffic participants (see also [Section 3.2.2](#)).

In general, the reference velocity profile is computed as the minimum of the speed limit obtained from the semantic road map and a reference speed limit that has been set as a property of the global planner. Furthermore, in curves the velocity profile is additionally adapted based on the curvature of the global path by interpolating between a given minimal and maximal velocity.

3.1.1 Re-planning of global path

The global reference path will be re-computed whenever a parking spot is already occupied. Such a situation is detected by the parking planner that executes the current parking task. An example of this global re-planning process for a blocked parking spot is illustrated in [Figure 3](#). Here, the vehicle follows the global reference path until the designated parking spot is reached. The local planner reports the successful completion of the navigation to the parking spot. Then, the mission executive, acting as a coordinating instance, triggers the parking task. The parking planner is unable to obtain a collision-free path into the parking spot and consequently triggers a *RoadGraphUpdate* message. A new parking spot is assigned by the server, and the global planner is triggered again (with an updated roadgraph hence yielding a different solution).

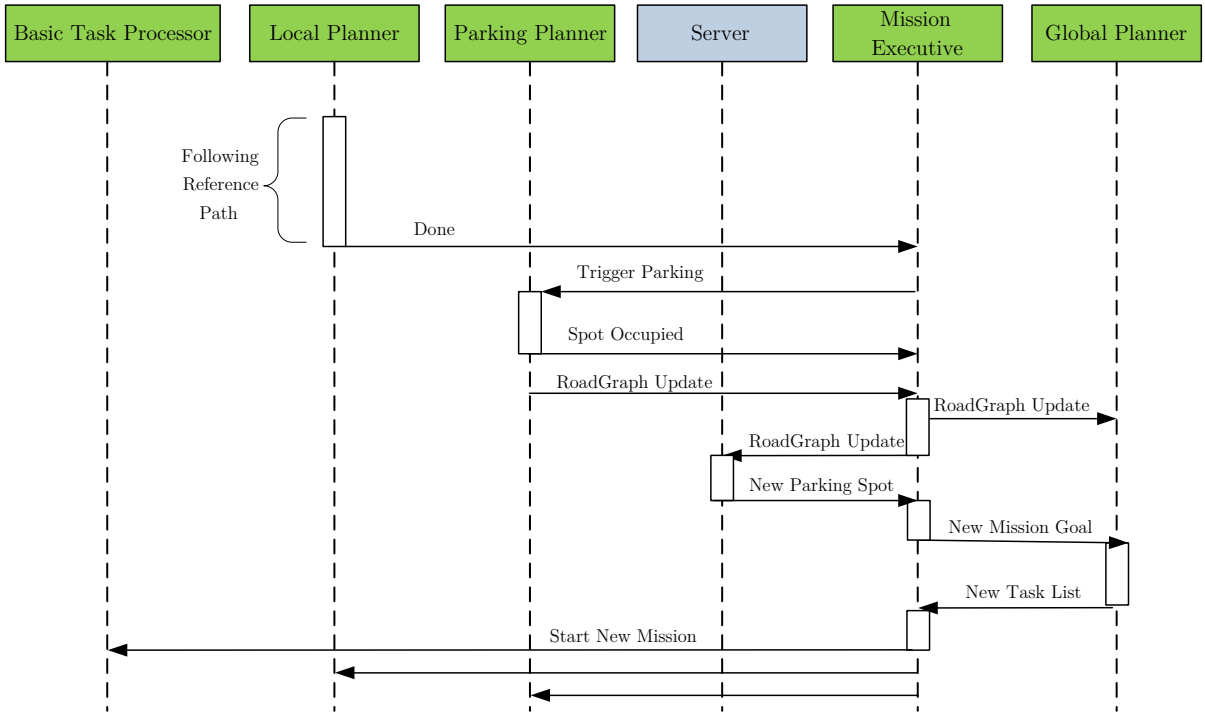
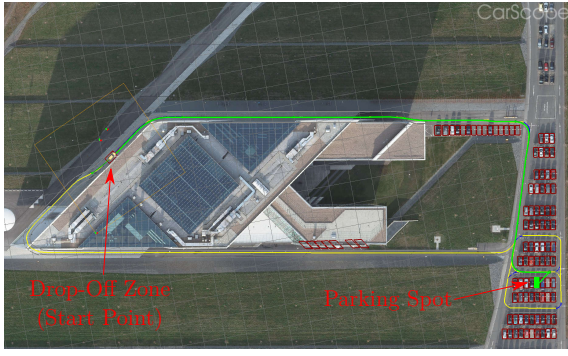
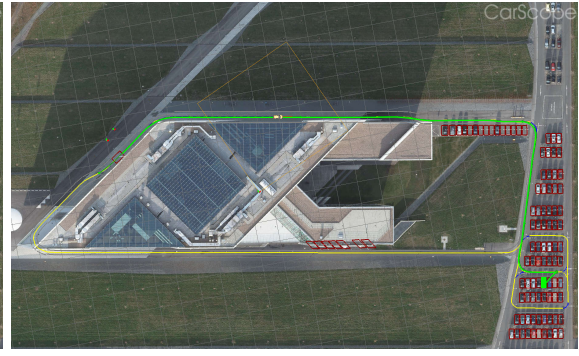


Figure 3: Sequence diagram, showing the signal flow when the parking spot is occupied.

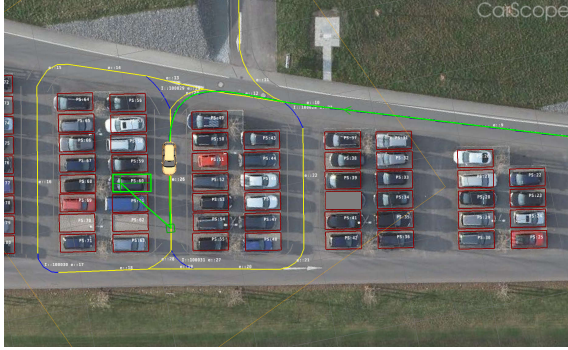
A simulated example of the global task planning process can be seen in [Figure 4](#): The user parks the vehicle in front of the Mobile Life Campus (MLC) in Wolfsburg. Then, the parking lot manager assigns a parking spot to the vehicle and the global task planner plans a sequence of tasks that lead the vehicle to the desired parking spot. The vehicle follows the planned trajectory.



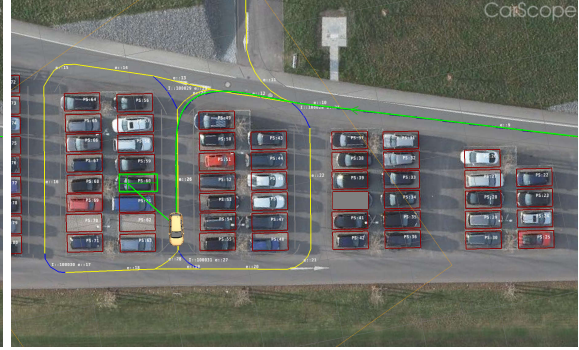
(4.a) Global path (green line) from drop-off zone (left) to parking spot (right).



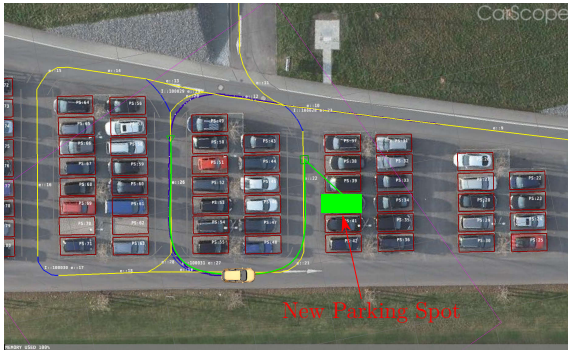
(4.b) The automated vehicle follows the trajectory.



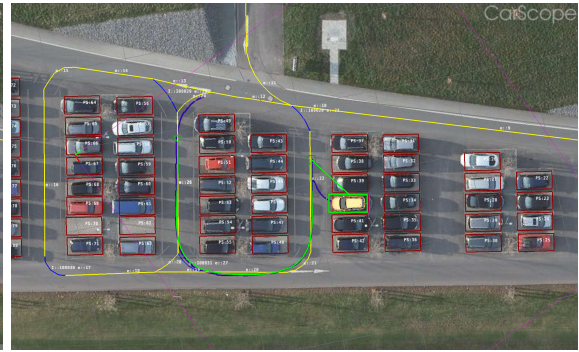
(4.c) The vehicle approaches the parking spot.



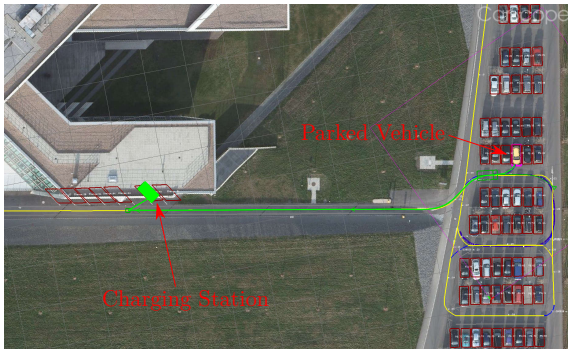
(4.d) The parking spot is occupied → re-planning.



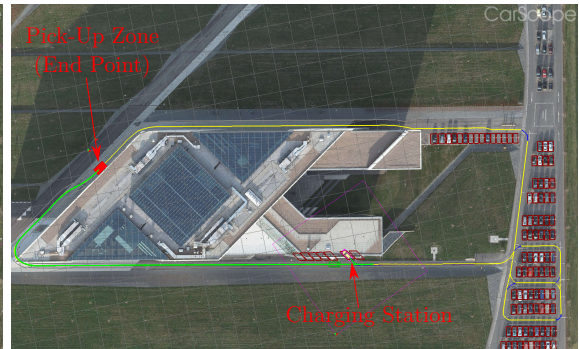
(4.e) The vehicle proceeds to a new parking spot.



(4.f) The parking task has been executed.



(4.g) After a trigger from the server, a new global path leads the vehicle to a charging station.



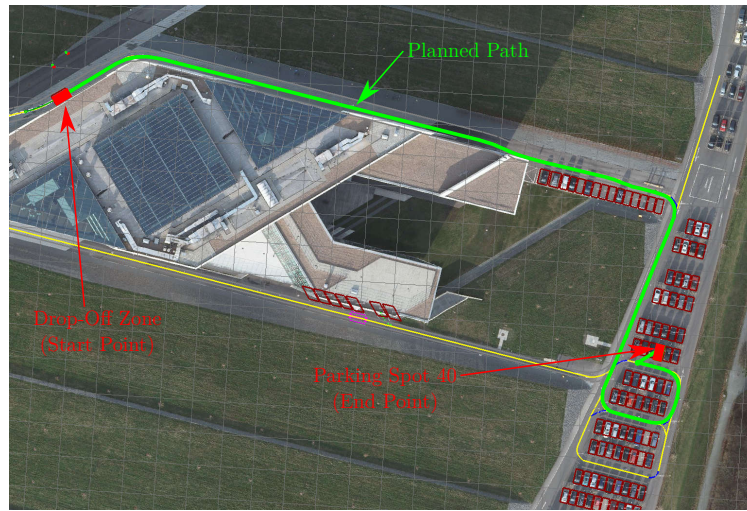
(4.h) After charging, another global path returns the vehicle to the pick-up point in front of the MLC.

Figure 4: Steps of a simulated parking/charging cycle at the MLC in Wolfsburg.

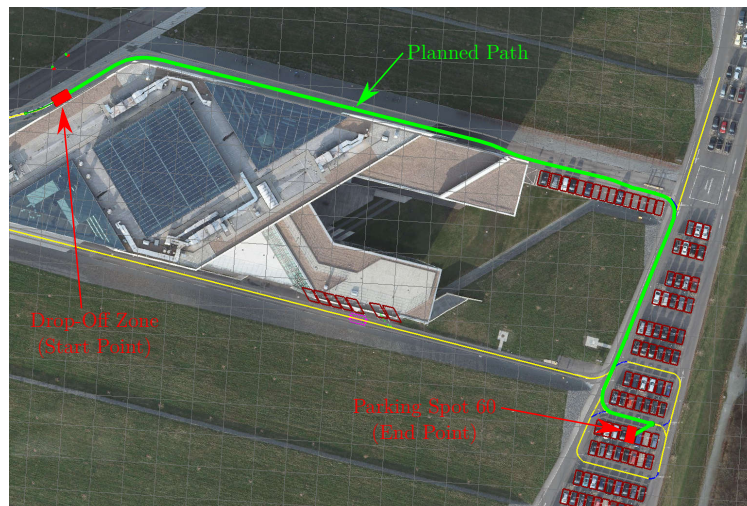
At the parking spot, the vehicle signals that the parking spot is occupied, which leads to a re-planning of the global path to another parking spot. The newly assigned parking spot is recognized as free so that a parking task is executed. After a trigger from the server – signaling that it is time to charge the vehicle – the global task planner plans a new sequence of tasks that lead the vehicle to the charging station. When the vehicle is fully charged, the global task planner plans a final task sequence that leads the vehicle back to the pick-up point at the front door of the MLC.

3.1.2 Computation time

The computation time of the global task planning module has been evaluated for the two scenarios that are shown in Figure 5. The goal of the depicted scenarios has been to drive automatically from the front door of the MLC in Wolfsburg to a parking spot located on the MLC’s car park, followed by a parking maneuver in this spot. The global task planner has been run for five times on each scenario, resulting in average execution times of 247 ms and 218 ms, each having a standard deviation of 8 ms. These Computation times include both the involved graph search as well as the smoothing step afterwards.



(5.a) Path (green line) from the MLC’s front door to the parking spot with ID 40.



(5.b) Path (green line) from the MLC’s front door to the parking spot with ID 60.

Figure 5: Evaluated parking scenarios: The automated vehicle is started at the MLC’s front door and drives automatically to a parking spot located on the MLC’s car park.

3.2 Online Partial Motion Planning (ETHZ-ASL)

The local on-lane motion planner is responsible for the navigation of the vehicle in the vicinity of the global plan acquired by the global task planner. Its functionality has been described in deliverables D 4.1, D 4.2, and D 4.3 as well as in the corresponding published papers [Schwesinger et al., 2013] and [Schwesinger et al., 2015a] attached to these deliverables.

The major changes and additions for the local motion planning framework since the last report can be summarized as follows:

- Navigation in tight spaces, e.g. the narrow gates at the MLC in Wolfsburg, has been achieved via a modification of the cost function to remove the high-cost area in narrow locations. The approach is detailed in [Schwesinger et al., 2015b] (Attachment A).
- Interaction with pedestrians has been enabled. The approach utilizing constant velocity predictions is detailed in [Schwesinger et al., 2015b] (Attachment A).
- Handling of intersections is supported via a scripting approach. Details of the approach can be found in Section 3.2.2.
- Semantic information is incorporated for vehicle movement prediction (see Section 3.2.3).

The following sections will detail functionality not yet covered by the last deliverables in workpackage 4.

3.2.1 Semantic Map Usage

Various modules from the vehicle’s local navigation stack such as the partial motion planner, the object prediction module, and the intersection handler utilize semantic information provided by the University of Oxford in workpackage 2 and address the specifications from milestone 6 for task 4.4. A commonly performed operation on the semantic map is a closest-point lookup of a given 2D pose to any lane in the semantic map. For a fast and precise computation of this lookup, lane information is being upsampled with the help of a spline interpolation during the module initialization. Additionally, a kd-tree data structure in either $\mathbb{R}^2$ configuration space (x/y) or in $\mathbb{SE}_2$ ($x/y/\theta$) is used. In $\mathbb{SE}_2$, a custom distance metric is employed as detailed in [Schwesinger et al., 2015b] (Attachment A).

3.2.2 Intersection Handling

At intersections, the majority of dynamic objects is located outside the field of view (FOV) of the stereo camera. Thus, objects have to be detected from the monocular cameras. The mono object detection framework (see deliverable D 3.5) requires the ego vehicle to be stopped at intersections. This necessary behavior of the ego vehicle has been implemented via a behavior scripting approach with different task processors (see deliverable D 4.1) involved. Though the local motion planner would in general be able to handle the intersection scenario on its own, the behavior scripting approach has been chosen to satisfy the stop-requirement imposed by perception.

The information about the location and relevance of intersections is extracted at the global planning stage. At this stage, the mission is already split into sub-tasks like `FollowTrajectory` and `PassIntersection`. The local motion planning framework handles the safe navigation up to the intersection stop line. A dedicated intersection task processor takes over at this point, monitors the intersection, and forwards control to the next task processor as soon as the intersection is considered to be free.

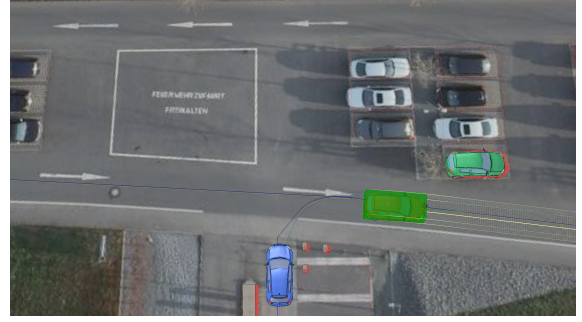
To determine the state of the intersection, any detected object is matched to a lane in the roadgraph. The ego vehicle will wait at the intersection if there is at least one object which matches all of the following conditions:

- The object’s speed is larger than $\|v\|_{\min}$.
- The Euclidean distance $\|\mathbf{p}_{\text{object}} - \mathbf{p}_{\text{ego}}\|_2$ in $\mathbb{R}^2$ to the ego vehicle is smaller than δ_0 .
- The object’s Euclidean distance to the roadgraph $\|\mathbf{p}_{\text{object}} - \mathbb{G}\|_2$ is smaller than δ_1 (roadgraph matching condition), where $\mathbb{G} \in \mathbb{R}^2$ represents the roadgraph.
- The object is located on an incoming lane of the intersection.

Additionally, the intersection handler requires the intersection to be free for 5 s so that the ego vehicle will continue its mission after waiting for 5 s at the free intersection. This verification procedure is illustrated in Figure 6.



(6.a) The ego vehicle (blue) waits at the intersection since the oncoming vehicle from the left (red = relevant object at the intersection) matches all conditions, including the check for the incoming lane.



(6.b) The crossing vehicle (green = non-relevant object at the intersection) is matched to an outgoing lane and the ego vehicle (blue) will continue its mission after waiting for 5 s.

Figure 6: Simulated scenario illustrating the semantic roadgraph verification step at intersections.

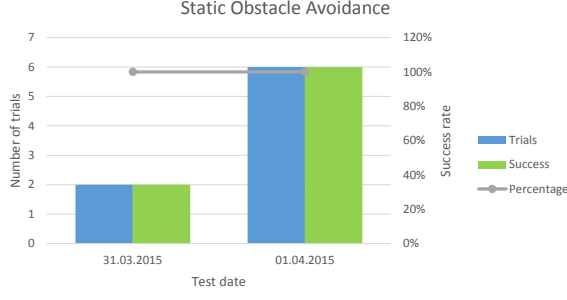
3.2.3 Vehicle Movement Predictions

In addition to the vehicle movement predictions utilizing lane information – detailed in [Schwesinger et al., 2015b] (see Attachment A) – semantic information is further used to deal with noise in the velocity estimates of the detected dynamic objects. For low-speed objects detected on parking spots, the velocity information is updated to be zero.

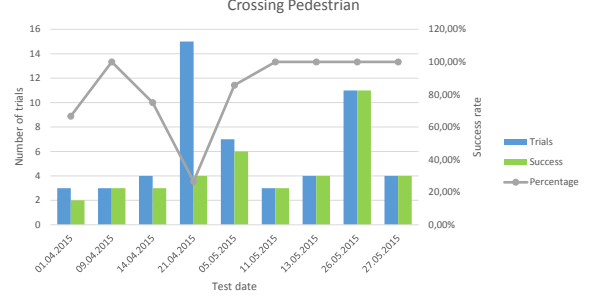
For a fast computation of parking spot associations, an axis-aligned bounding box (AABB) tree data structure is used, equivalent to the data structure used for fast collision queries with dynamic objects (see [Schwesinger et al., 2015a] and deliverable D 4.3). The geometric parking spot descriptions are inserted into the AABB tree data structure, allowing for a fast computation of the penetration depth with the detected dynamic objects. A dynamic object is successfully associated with a parking spot if $d_p \geq \min(l_o, w_o)/2$, with d_p being the penetration depth and l_o , w_o being the length and width of the object, respectively.

3.2.4 Evaluation

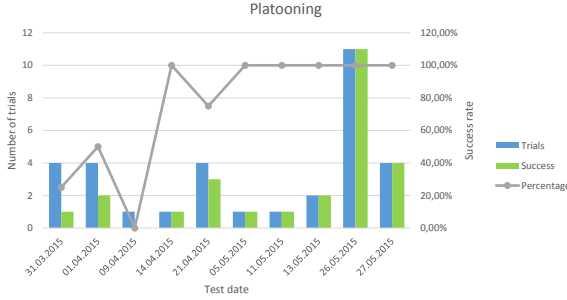
In this section, evaluation results for the local motion planning framework, operating in different scenarios, is shown. On the one hand, statistics have been retrieved via an automatic log parser. This evaluation procedure offers the advantage that a statistically relevant amount of data can be inspected. Yet, the automatic interpretation of the correctness of the ego vehicle’s behavior is impossible. On the other hand, manual statistics during defined tests for individual maneuvers have been created in the run-up preparations for the MLC demo on the 25th of July, 2015. These manual statistics give information about the correctness of the behavior of the ego vehicle in the investigated situations. However, it is extremely time-consuming to collect a statistically relevant amount of test data.



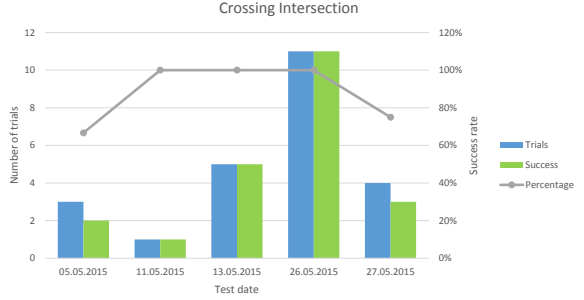
(7.a) Statistics for static obstacle avoidance scenario.



(7.b) Statistics for crossing pedestrian scenario.



(7.c) Statistics for platooning scenario.



(7.d) Statistics for intersection scenario.

Figure 7: Manual statistics recorded for tests of specific maneuvers. A successful test includes general faultless as well as appealing behavior from a human observer’s point of view. The x-axis represents the test days and the y-axis represents the total/successful number of tests (left side) and the percentage of successful tests (right side), respectively.

Manual Evaluation in Different Scenarios Figure 7 shows manually collected statistics for four different scenarios. Static obstacle avoidance proved to be reliable from the early beginning, rendering continuous integration tests superfluous (Figure 7.a). The low stereo camera stixel count for pedestrians gives good estimates of their object centers and therefore low-noise velocity estimates. This is reflected in an overall good navigation performance in the presence of crossing pedestrians. Except for one test day that included a software bug in the dynamic object detection module which suppressed the output of objects, the success rate of this scenario has been 100% at the final integration stage (Figure 7.b). Platooning behaviour has been implemented since the Stuttgart demonstration in April 2014 and shows flawless operation in the absence of perceptive shortcomings. As explained in deliverable D 3.5, the separation of static and dynamic content in the scene occasionally suffers from missing depth information in the center of the preceding car, leading to portions of the car appearing in the static gridmap. The reaction of the local planning framework to static obstacle information in front of the ego vehicle is prompt. Hence, avoidance maneuvers are initiated immediately upon these events (Figure 7.c). The approach taken to handle intersections is straightforward and the vehicle’s operation is only subject to the results from the mono object detection as detailed in deliverable D 3.5 (Figure 7.d).

Automated Evaluation Figure 8 visualizes statistics that have been automatically generated from 471 GB of log files, recorded over the course of 6:47 hours of automated operation in which 24.63 km have been covered. A detailed explanation of those statistics can be found in Attachment A.

Figure 9 shows situations that often provoked events in which no solution could be found by the local planner (hence triggering an emergency brake). These events stem firstly from pedestrians moving in very close proximity to the vehicle. In these situations, the purely reactive planning approach is over-conservative. A cooperative prediction scheme would remedy this shortcoming and is among our current research goals. Secondly, these events sporadically occur when passing through narrow gates on the parking lot with only a few tens of centimeters margin on each side of the vehicle – a highly challenging scenario for both vision-based perception and motion planning.

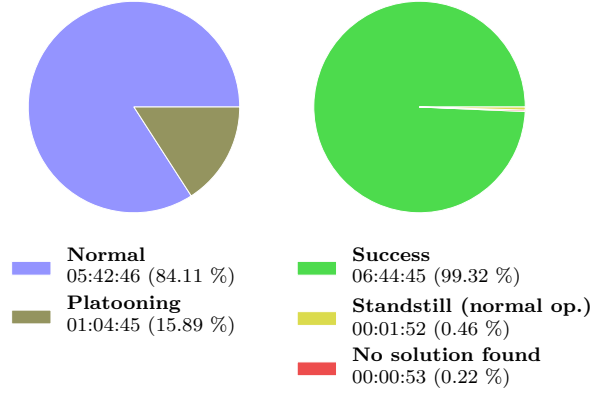


Figure 8: Local motion planning operation mode (left) and failure statistics (right) generated from 24 test days over the course of two month.



Figure 9: Moving pedestrians in close vicinity to the ego vehicle (left) and small perception noise when passing narrow gates (right) were often the cause when no valid solution trajectory could be computed by the local motion planner.

Smoothness of Steering Motion For lateral control, The V-Charge vehicles adopt the interface of the 'Park Assist' which only provides methods to request a specific steering angle to the vehicle. The steering speed at which the requested angle is approached is the maximum steering speed. This fact renders it difficult to achieve smooth steering behavior, see [Galceran et al., 2015].

Figure 10 depicts simulated steering angle profiles of the solution trajectory computed by the local planner, as well as measured and commanded steering angles. The simulated profiles represent the optimal steering angle command sequence over the next 10 seconds, computed at a certain planning cycle. The future values are never actually applied to the platform, since replanning takes place in the next planning cycle. The step changes in the plot occur due to unforeseen necessary reactions to obstacles. The first three trajectories do not “see” these upcoming reactions yet, hence miss the corresponding steps in their profiles. One can observe that the steering angle profiles computed within each planning cycle are fairly smooth. Steering jerks occur upon reaction to obstacles. Besides that fact, the prediction of the system input is very stable across replanning cycles in the absence of obstacles, indicating a good internal model for the transfer function.

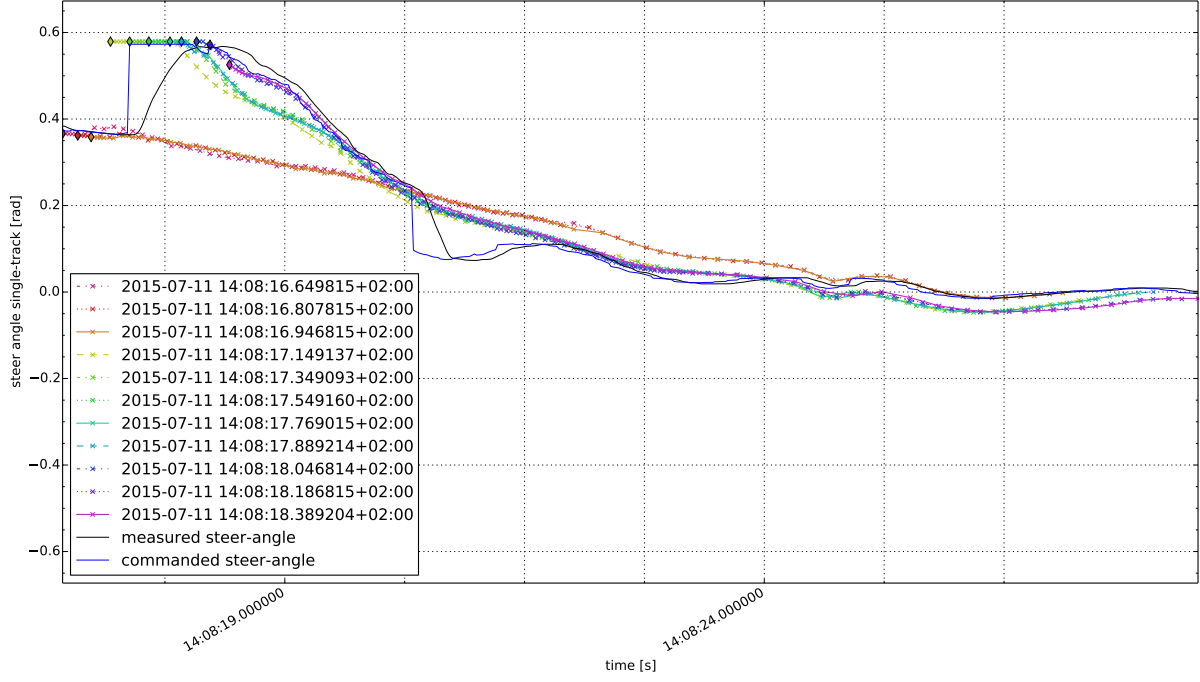


Figure 10: Simulated steering angle profiles of the solution trajectory computed by the local planner, as well as measured and commanded steering angles.

With a steering speed/torque interface at hand, the planner would be able to adapt the steering speed to the criticality of the situation. Additionally, small (in amplitude) high-frequency steering oscillations due to discretization effects in the planner could be almost completely damped.

Navigation at Higher Speeds The default reference speed we used in all experiments is 1.6 m/s or 5.76 km/h respectively, which is still far below the theoretically possible maximum speed of 10 km/h . This choice has purely practical reasons. With that speed, safety personal and visitors are able to conveniently follow the vehicle by foot. Experiments with a higher reference speed of 2.22 m/s or 8 km/h respectively were performed (see Figure 11) without any negative side effects on the systems' performance. On the straight segments of the parking level, the car reaches the requested speed, while in turns it automatically reduces its speed to increase comfort.

Tracking Error The controller utilized by the local on-lane planner mainly extends the longitudinal vehicle behaviour via a PI (proportional and integral) acceleration control loop. This enables handling of slopes as shown in deliverable 4.3 in the Bosch parking garage, Stuttgart and in the multilevel tests at Excellence Parking, Amsterdam. The lateral system dynamics are mainly untouched, due to the fact that the local on-lane planner recomputes a feasible solution roughly each 100 ms starting at the current vehicle state and effectively resetting the error states in every planning cycle. Apart from the aforementioned longitudinal acceleration control loop the controller is thus almost inactive during on-lane operation, given that the local on-lane planner already computes a sequence of system inputs for the platform.

An evaluation of the local on-lane planner with respect to its tracking capabilities of a global reference path is not particularly meaningful, as the local on-lane planner performs a multi-objective optimization trading costs for deviation from the reference path against other optimization criteria such as deviation from a reference velocity and distance to static and dynamic objects. Yet we want to present data for the tracking error of a global reference path, partially in the presence of obstacles. The tracking performance in the absence of obstacles is partially a result of the tracking capabilities of the forward-

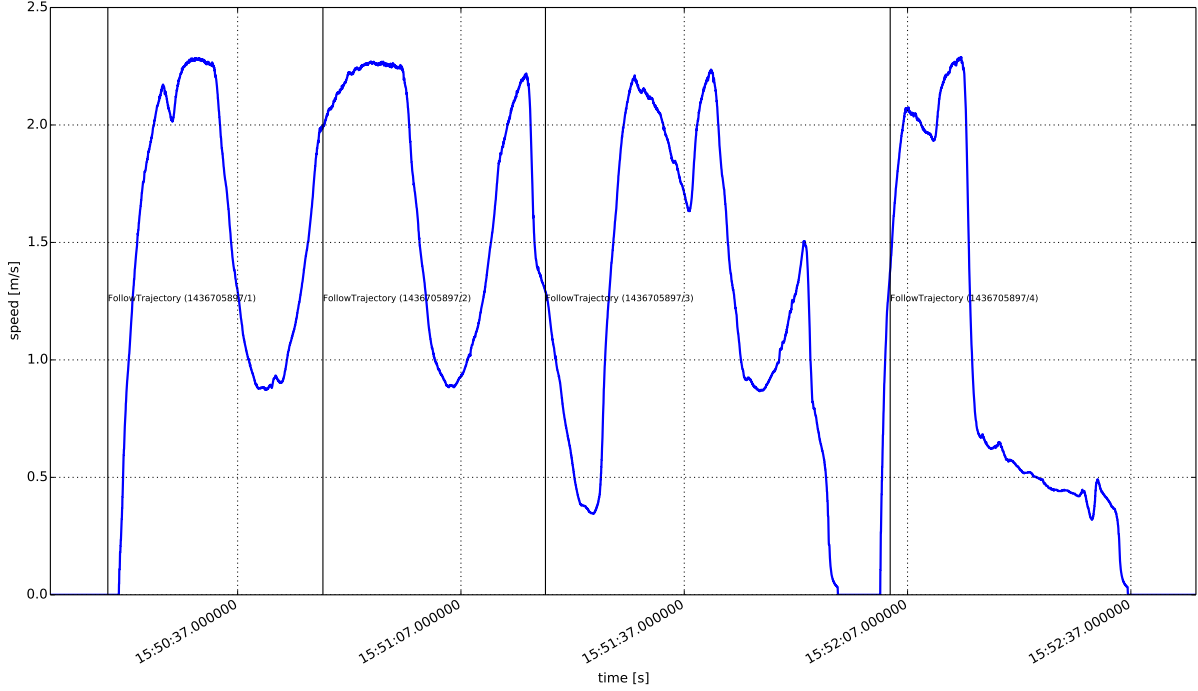


Figure 11: Speed profile during one test run at Schiphol Excellence Parking with a reference speed of 8 km/h.

simulated controller in the local planning framework, regulating a model of the system towards samples drawn from a state manifold depending on the reference path. Note that neither the forward-simulated controller nor the local on-lane planner was ever tuned for tracking performance. The controller was even modified with a deadband region in path distance to yield smoother steering behavior, sacrificing part of the tracking quality.

Figure 12 depicts the tracking error of the local planner for a single run in the Excellence Parking Garage in Amsterdam. In both curves and straight segments, the maximum tracking error is approximately 20 cm, fulfilling the specifications from milestone 6 for task 4.3. The median tracking error for the full run is 0.11 cm. The oscillating behavior can be attributed to the deadband in the simulated controller, which was introduced in the preparation phase of the final demonstrations. The peak at the beginning of the run is caused by an initial offset of the vehicle, the peak at 15:51:55h from a reaction to a crossing pedestrian.

3.2.5 Conclusion

The local motion planning framework has proven to be stable, reliable, and an appropriate choice for static scenarios and in the majority of cases also for dynamic ones. The framework is very well scalable via adjusting the number of samples. The purely reactive planning approach, neglecting the reaction of other traffic participants to our actions, shows promising results also for dynamic scenes. Yet, interaction-aware planning techniques for close-range interaction with pedestrians would improve navigation performance. ETHZ-ASL considers them mandatory to avoid overly conservative behavior during such interactions. ETHZ-ASL is further pursuing this research direction in the European project EUROPA2, grant agreement no 610603.

To improve the smoothness of navigation, ETHZ-ASL recommends to provide a steering speed interface instead of a steering angle interface to the vehicle. This would enable the planner to choose between different aggressiveness levels of maneuvers which would result in a smoother navigation performance.

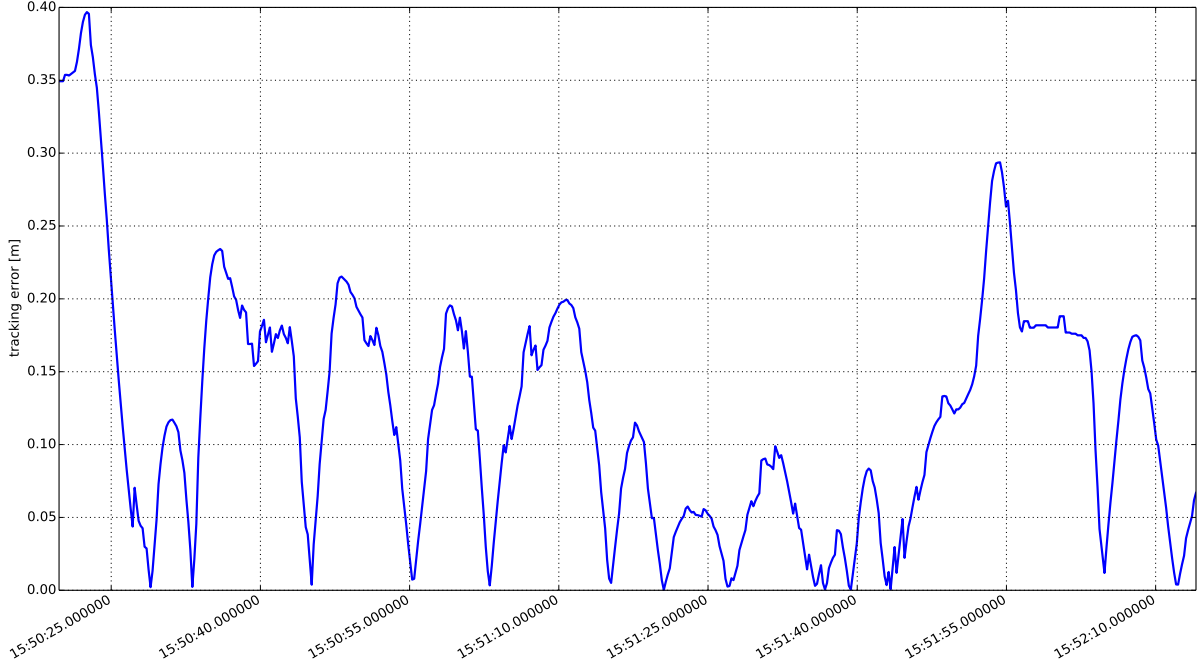


Figure 12: Tracking error of the local planner for a single run in the Excellence Parking Garage in Amsterdam driven with 8 km/h.

3.3 Off-Lane/Parking Planner (VW)

The parking task processor is the module responsible for executing forward and backward parking maneuvers as well as the respective parking out maneuvers. In the context of the V-Charge project, the process of parking involves the following tasks:

- Choice of a parking spot / charging station
- Determination of the desired final vehicle pose within the parking spot (goal pose)
- Computation of a local trajectory to the goal pose
- Detection of the occupancy state of the parking spot
- Execution of the trajectory
- Reaction to changes in the occupancy state, position, orientation, and size of the parking spot

The remainder of this subsection is structured as follows: we first offer a high-level description of the most important aspects of the approach, then we provide the evaluation results, and finally we conclude the section and give an outlook to future work.

3.3.1 Overview of the Approach

In the following, we provide a short description of the main tasks from the parking process.

Choice of a parking spot In the V-Charge setting, the parking lot management systems assigns a parking spot / charging station to the car. It is also possible to determine the desired final vehicle pose within the parking spot based on local sensor data.

Determination of the desired final vehicle pose within the parking spot Errors in vehicle position estimation can impact the accuracy of the parking maneuver. To address this issue, two different algorithms have been developed. They have been described in deliverable D 3.5. The goal of all those algorithms is to refine the available transformation from the target parking spot to the ego vehicle’s pose.

The two algorithms from deliverable D 3.5 can manage two different types of parking spots. The first one is able to detect “Tile Grid” parking spots as shown in Figure 13.a, i.e. parking spots that show some type of cobblestone pavement, and the second one is able to identify “White Stripes” parking spots as depicted in Figure 13.b, i.e. parking spots that are bounded by white stripe lane markings. Both algorithms have been evaluated in deliverable D 3.5 by measuring the relative angle between the parking spot and the vehicle and by comparing it with the angle obtained from the two algorithms. The “Tile Grid” parking spot algorithm yields an average error of 0.11° and the “White Stripes” parking spot algorithm yields an average error of 0.54° , respectively.



(13.a) “Tile Grid” algorithm



(13.b) “White Stripes” algorithm

Figure 13: Algorithms for parking spot refinement.

Computation of a local trajectory into the goal pose Once the pose of the desired parking spot has been determined, the parking planner computes a local trajectory into the goal pose. The objectives for a local trajectory are:

- The trajectory has to be collision-free.
- The trajectory has to lead the car into the desired goal pose (or very close to it).
- The trajectory should look natural, i.e. it should contain only a minimal number of direction changes and it should use only the necessary maneuvering space.

In order to achieve these goals, the parking planner differentiates between backward and forward parking maneuvers. We will first describe the backward parking maneuver and then proceed to the forward parking maneuver.

Backward parking: For the backward parking task, the online partial motion planner hands over the task execution to the parking task processor via the mission executive after having passed the desired parking spot. At first, the parking task processor stops the vehicle. Then, the local trajectory into the goal pose is computed – under consideration of static obstacles from the grid map – with the help of the following planning cascade:

1. Try the *Reeds & Shepp Planner*.
2. If step 1 failed: Try the *Pseudo Deterministic Planner*.
3. If steps 1 and 2 failed: Try the *A-Star Planner*.

The parking planner iterates through these three steps until one of the algorithms finds a valid and collision-free trajectory. The collision check is performed by advancing a vehicle model along the discretized trajectory and looking up the static obstacles from the local grid map. The three algorithms have different characteristics which can be summarized as shown in Table 3:

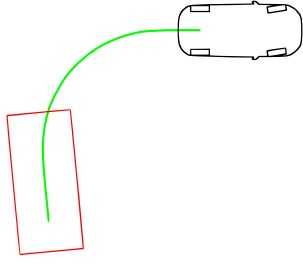
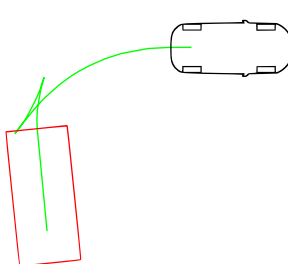
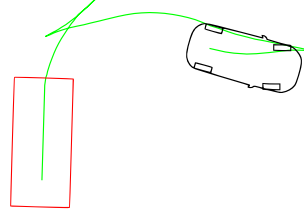
<i>Reeds & Shepp Planner</i>	<i>Pseudo Deterministic Planner</i>	<i>A-Star Planner</i>
• Based on Reeds & Shepp curves [Reeds and Shepp, 1990] • No change of direction • One single move consisting of two straight line segments and a circular arc 	• Similar to the <i>Reeds & Shepp Planner</i>, but: • One single change of direction • Three moves consisting of two straight lines and three circles 	• Free space planner • Undefined number of direction changes • Multiple moves consisting of predefined motion primitives 

Table 3: Characteristics of the different backward planning algorithms.

Forward parking: In a forward parking task, the parking planner starts to permanently compute local trajectories from the current vehicle pose into the target parking spot when the vehicle is located at a certain distance from the desired parking spot. The computation ends once a collision-free trajectory has been found. At this point, the parking task processor signals readiness and the task execution is handed over from the local planner to the parking task processor. The parking task is then performed in one single move without previously stopping the vehicle. There is a planning cascade with increasing complexity and time costs similar to the backward parking task:

1. Try the *Clothoid Planner*.
2. If step 1 failed: Try the *Reeds & Shepp Planner*.
3. If steps 1 and 2 failed: Return to step 1.

Like the *Reeds & Shepp Planner*, the *Clothoid Planner* does not allow direction changes. The resulting local trajectory consists of two straight line segments that are connected by a clothoid, a curve with linearly increasing curvature. Such a curve is used in the forward parking scenario, instead of the Reeds & Shepp curves, because the vehicle is not stopped before the parking process so that a trajectory which exhibits minimal lateral force should be used in order to maximize driving comfort.

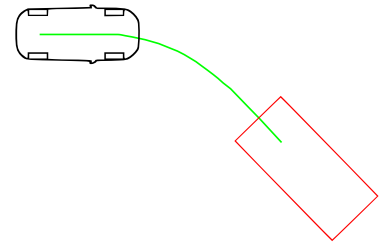


Figure 14: Clothoid Planner.

If no valid forward parking trajectory has been found until the vehicle has passed the parking spot, the vehicle is stopped and the *A-Star Planner* is used to find a valid parking trajectory. The resulting trajectory, if found, is expected to be very complex and to consist of mixed backward and forward motions.

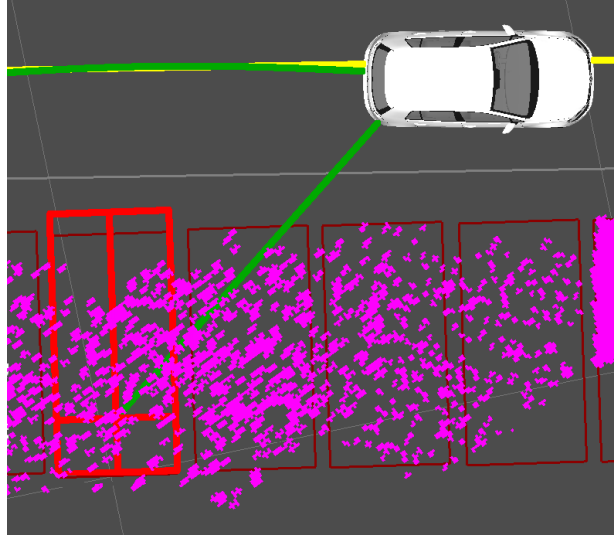


Figure 15: Simulated example of an occupied parking spot (red) that caused the abortion of the parking task because no collision-free local trajectory could have been found by the parking task processor. Obstacles from the local grid map are shown in magenta, the road graph is shown in yellow, and the global path is shown in green, respectively.

Detection of the occupancy state of the parking spot As mentioned above, the parking task processor tries to find a collision-free local trajectory into the parking spot under consideration of static obstacles from the local grid map. This may fail because a parking spot can be occupied by another car or because there can be some obstacles that block the automated vehicle’s way to the parking spot (i.e. the space in front of the parking spot can be occupied).

Within V-Charge, there exists no separate module to check the occupancy state of a parking spot. Instead, the occupancy state is a direct result from the above-mentioned planning cascade: A parking spot is set to blocked when no collision-free local trajectory into the parking spot could have been obtained (see Figure 15). This is also signaled to the mission executive using a *Spot Occupied* message. Additionally, a *RoadGraph Update* message is sent. As a result, another parking spot is sent as a new mission goal.

Execution of the trajectory Once a feasible parking trajectory has been found, the automated vehicle tries to follow this trajectory into the parking spot. For this purpose, the trajectory controller that has already been described in deliverable D 4.1 is used.

The task of the trajectory controller is to execute a reference trajectory with best-possible precision. The module serves as an abstraction layer to low-level vehicle control and transmits longitudinal acceleration and an absolute steer-angle. The longitudinal and lateral control schemes are separated from each other and thus reduce control complexity substantially. The employed lateral control scheme operates in analogy to a charged particle within a dipole’s electric field, where the steer angle is obtained from the curvature of the field lines. The longitudinal control interprets reference speeds along the trajectory, obtained from a pre-processing step, using four states: *Launch*, *Acceleration*, *Steady*, and *Brake*. See deliverable D 4.1 for more details on the trajectory controller.

The only minor adaptation of the trajectory execution which has been made in the 4th period is that a creep mode has been added in the pre-processing step: When the vehicle is close (1 m) to the desired target pose, the desired reference velocity is strongly reduced. This creep mode has a very positive effect on the achieved parking accuracy. However, it also has the effect that it looks like the vehicle shortly stops before it reaches the final parking pose. A smooth interpolation between the normal trajectory following mode and the creep mode would possibly solve this problem.



Figure 16: Daily parking success rate for the time period between 2nd April, 2015 and 16th June, 2015.

Reaction to changes in the occupancy state, position, orientation, and size of the parking spot During the execution of the trajectory, irregularities can occur that have not been considered in the planning process, e.g. changes in the occupancy state of the parking spot.

After having reached the final parking position, there may be an inaccuracy in the parking pose because the parking trajectory has been planned only once before the execution of the parking task and newly arrived parking spot hypotheses after this point have not been considered. However, as the parking spot hypotheses tend to become more accurate when the vehicle enters the parking spot because more sensor data becomes available, e.g. from the ultrasonic sensors, there is an option to correct the final parking position with the help of this new parking spot hypotheses. More precisely, the parking task processor is able to perform a correction move to achieve a better parking result. To do so, the vehicle pulls back out of the parking spot and moves back in right away whilst performing some longitudinal and lateral corrections.

Reaction to changes in the occupancy state of the parking spot, e.g. when some pedestrians or cars suddenly appear, have not been explicitly implemented within the parking task processor. However, if the distance between these dynamic obstacles and the ego vehicle becomes too small, the collision watchdog, based on ultrasonic sensors, acts as a last line of safety and triggers an emergency stop (see Figure 1). In case of an emergency stop and a continue of the mission afterwards, the global path and the local parking trajectory remain the same.

Changes in the size of the parking spot, e.g. when the width of the parking spot has been overestimated so that neighboring cars get too close during the execution of the parking task, have also not been explicitly addressed within the parking task processor. Still, if the distance between the parked cars and the ego vehicle becomes too small, the collision watchdog again triggers an emergency stop. In this case, however, the current mission has to be aborted because a continue of the mission would require a recomputation of the parking trajectory which has not been implemented.

3.3.2 Experimental Evaluation

In Figure 16, one can see the great robustness of the developed parking approach. In total 267 documented parking manoeuvres (backward and forward parking manoeuvres) have been performed over a time-

period of about 2 months. On average, a parking success rate of 97% could be obtained. However, it can also be seen that on many days the parking success rate has reached a full 100%.

The accuracy of the parking manoeuvres has been evaluated on two different scenarios (one forward parking and one backward parking scenario). Additionally, the parking success rate for a varying width of the parking spot has been evaluated in a simulation.

Forward parking scenario

Experimental Setup:

For the evaluation of the forward parking scenario, the vehicle has first been parked manually into the desired parking spot. This pose has then been used as the ground truth pose by defining a reference point at the vehicle's front and saving the coordinates of this point (in the coordinate system of the semantic map). Afterwards, the vehicle has been pulled back out of the parking spot and positioned in an appropriate distance from the parking spot. Then a local forward parking trajectory into the parking spot has been planned and the automated parking process has been executed. This has been repeated several times and for each parking task the deviation of the reference point from the ground truth point has been noted. The final position of the vehicle has thereby been obtained with the help of the visual localization system (see also [Section 3.3.2](#)).

Results:

The results for the forward parking scenario are shown in [Figure 17](#). It can be seen that the forward parking process is very accurate. The mean absolute longitudinal deviation is 3.3 cm with a standard deviation of 2.1 cm, and the mean absolute lateral deviation is 2.1 cm with a standard deviation of 1 cm, respectively. The required accuracy for a successful charging maneuver is ± 15 cm in longitudinal direction and ± 10 cm lateral direction. So, the measured error is on average 11.7 cm lower in longitudinal direction and 7.9 cm lower in lateral direction than the defined maximally allowed error.

In addition to the final vehicle pose within the parking spot, we also evaluated the lateral deviation between the planned parking trajectory and the vehicle's trajectory during the forward parking task. An example for the planned and actually driven trajectory is shown in [Figure 18](#). It can be seen that the maximal deviation occurs towards the end of the turning right maneuver at position $(-5.6/0.5)$. A similar behavior can be observed for all evaluated parking scenarios. The recorded mean absolute lateral deviation, averaged over eight runs, is 8.2 cm with a standard deviation of 0.9 cm.

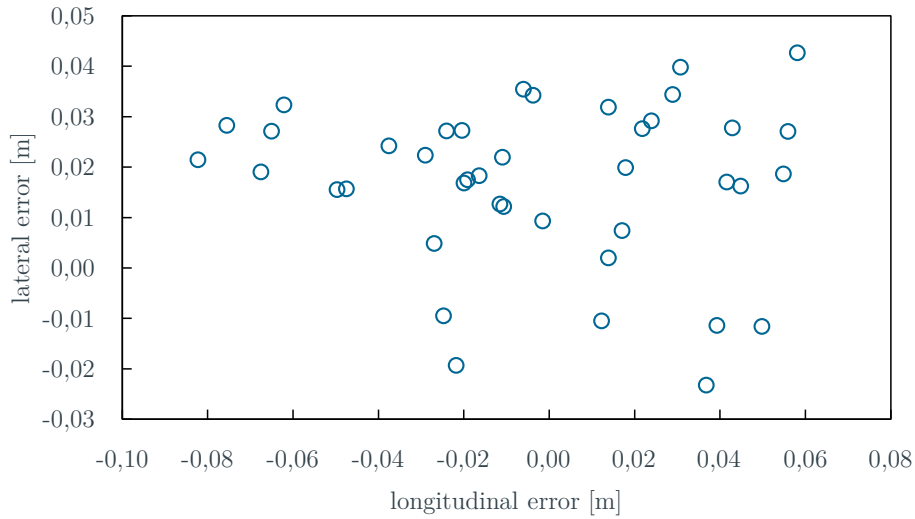


Figure 17: Resulting errors between the final vehicle pose and the ground truth pose for the forward parking scenario.

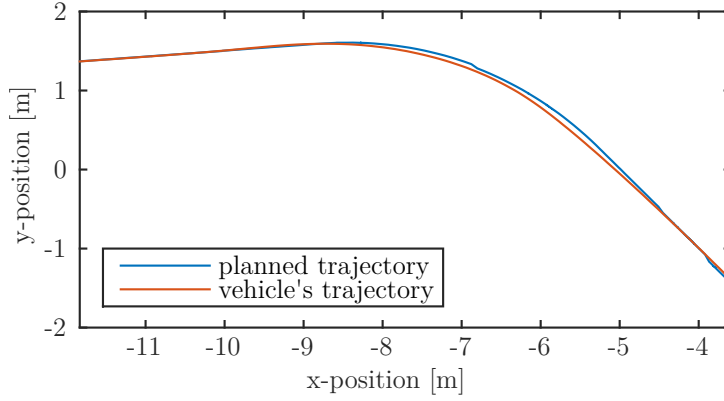


Figure 18: Example for a planned (blue) and actually driven (red) parking trajectory.

Backward parking scenario

Experimental Setup:

The setup of the evaluated backward parking scenario has been very similar to the setup of the forward parking scenario. The goal in this scenario has been to perform an automated backward parking task between two parked vehicles. For this purpose, the ground truth pose has again been obtained by parking the vehicle manually – this time backwards – into the desired parking spot, defining a reference point at the vehicle’s front, and saving the coordinates of this point (in the coordinate system of the semantic map). In order to perform the evaluation, the following steps have then been repeated several times:

1. Position the vehicle in an appropriate distance from the parking spot.
2. Plan a local backward parking trajectory into the parking spot.
3. Execute the parking task.
4. Write down the deviation of the reference point from the ground truth point.

Results:

The results for the backward parking scenario are shown in [Figure 19](#). It can be seen that the backward parking process is also very accurate. The mean absolute longitudinal deviation is 6.8 cm with a standard deviation of 6.5 cm and the mean absolute lateral deviation is 6.4 cm with a standard deviation of 4.6 cm, respectively. This means that the mean errors lie in the same order of magnitude as the errors of the forward parking process. However, the longitudinal error is on average twice as high and the lateral error is on average thrice as high, respectively. This is due to the fact that the backward parking task is a bit more complicated than the forward parking task.

Like in the forward parking task, it can also be seen that the longitudinal variance in the measured parking position is larger – almost twice as large – than the lateral variance. This is most probably due to the local trajectory following module:

When the vehicle is already close to the desired target pose – but not close enough – an acceleration command is triggered which causes the vehicle to overshoot the desired target pose.

The two lateral outliers that have been emphasized by red circles in [Figure 19](#) are caused by wrongly detected obstacles that forced the vehicle to choose a sub-optimal trajectory into the desired parking spot (see [Figure 20](#)). These false positives have been caused by the rear stereo cameras while passing the parking spot in forward direction. So, a quick workaround has been to disable the rear stereo cameras while driving forward and use them only when driving backward. However, this issue has to be further addressed in future work.

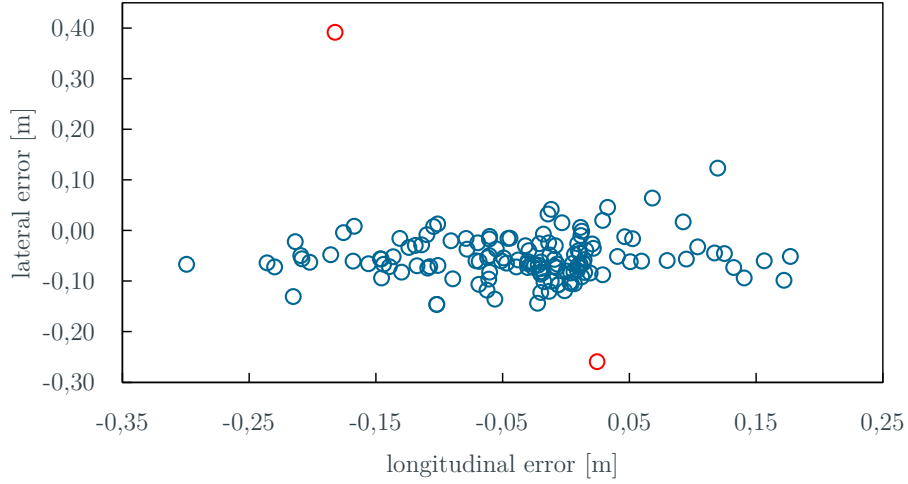


Figure 19: Resulting errors between the final vehicle pose and the ground truth pose for the backward parking scenario.

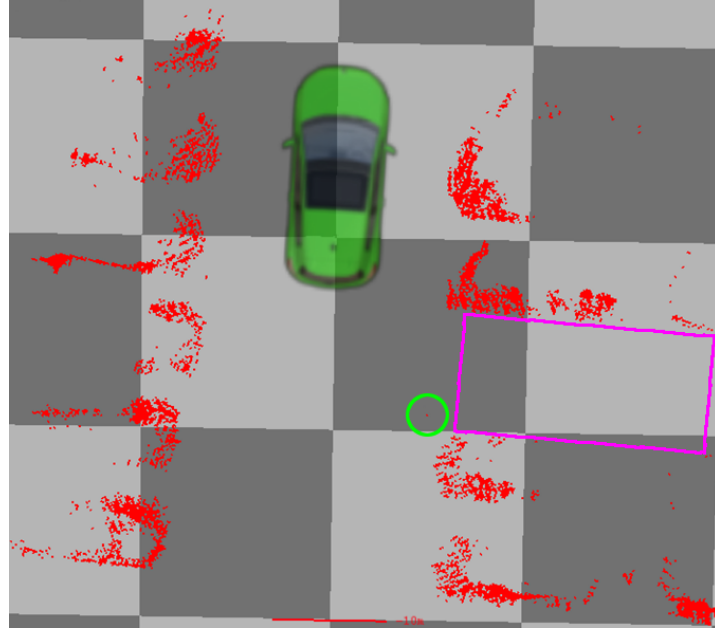


Figure 20: Wrongly detected obstacle (red point inside the green circle) – caused by the rear stereo cameras – that forced the vehicle to choose a sub-optimal trajectory into the desired parking spot (magenta rectangle).

Evaluation of different parking spot sizes

Experimental Setup:

For the simulated evaluation of the parking success rate for different parking spot sizes, static obstacles that represent neighboring vehicles (magenta regions in Figure 21) have been drawn manually into the local grid map at a cell-resolution of 5 cm. Afterwards, the desired parking pose has been reached by manually positioning the vehicle central between the obstacles (green vehicle in Figure 21). This process has been repeated for every simulated parking spot width. For the evaluation, the vehicle has then been positioned in an appropriate distance orthogonal to the parking spot and the parking task has been executed.

Similar to a real setup, the manually positioning of the vehicle causes a slight variation in the vehicle's start position. So, the simulated parking process has been repeated 20 times for each simulated width and it has been noted whether the parking process has been successful or not. In case of a successful parking process, the number of needed parking moves has also been noted. Unsuccessful parking tasks due to simulation errors have not been regarded, and the width of the parking spot has been obtained from the simulation environment by two mouse clicks in the area of the front wheels. The width of the simulated ego vehicle has been 2.2 m.

Results:

The evaluation results are shown in Table 4. It can be seen that for a sufficient wide parking spot, having a width of 2.64 m, the parking success rate is 100 %. Additionally, for this width the parking task could have almost always been completed within one single move. There has been only one task that needed three moves. For a decreasing parking spot width, the success rate gradually decreases and the average number of parking moves gradually increases, respectively. However, also for a tight parking spot, having a width of 2.28 m, the parking success rate is still as high as 75 % and on average only 2.25 moves are needed in order to perform the parking task.

width of simulated parking spot	average number of parking moves	success rate
2.64 m	1.10	100 %
2.44 m	1.50	90 %
2.36 m	1.53	80 %
2.28 m	2.25	75 %

Table 4: Parking success rate and average number of needed parking moves for a varying width of the parking spot and a slightly varied vehicle start pose.

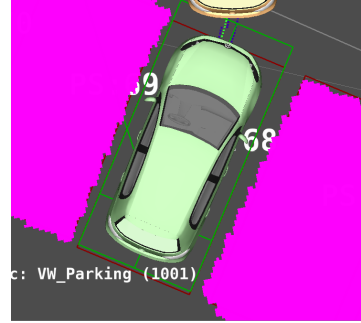


Figure 21: Simulated parking spot, having a width of 2.44 m.

Evaluation of the random visual localization error in the final vehicle pose

In Section 3.3.2 and Section 3.3.2, the final vehicle pose in the parking spot has been obtained via the visual localization system. So, the final vehicle pose contains a localization error. The absolute localization error has been thoroughly evaluated in [Mühlfellner et al., 2015]. However, for our parking experiments we need to determine the relative localization error between two spatially close poses: the desired parking pose and the reached parking pose, respectively.

In order to assess this error, we manually recorded the final vehicle pose inside the parking spot in addition to the pose that has been obtained via the visual localization system. For this purpose, we attached a plumb line at the vehicle's front and marked the reached position on a sheet of paper after each successful parking process (see Figure 22).

Afterwards, we removed the systematic error that originates from the translational and rotational offset between the paper sheet's origin and the desired parking pose. By doing this, we are able to evaluate the variance in the measured vehicle poses that originate from the visual localization system.

Figure 23 shows the manually measured and the visually located vehicle poses for a subset of the forward parking results from Figure 17. It can be seen that the manually obtained vehicle poses and the poses that have been obtained via the visual localization system strongly correlate with each other. In fact, the mean absolute deviation between the visual localization and the manual measurements is only 3 mm with a standard deviation of 1.9 mm. The random localization error is shown in figure Figure 24. It is clearly visible that the error shows a Gaussian distribution (one, two, and three standard deviations of the Gaussian distribution are shown in red, green, and blue, respectively).



Figure 22: Manual recording of the final vehicle pose inside the parking spot via a plumb line attached at the vehicle's front.

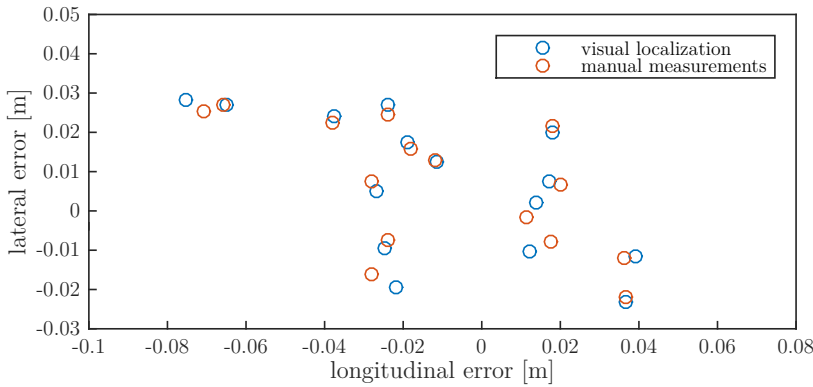


Figure 23: Comparison between the visually located (blue) and manually obtained (red) final vehicle poses within the parking spot. Note that the systematic localization error has been removed.

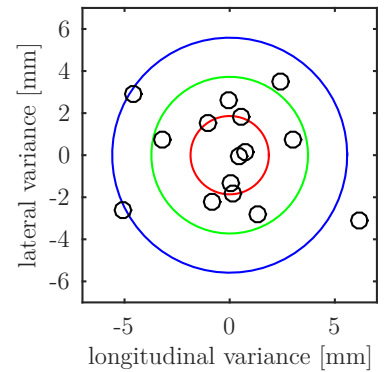


Figure 24: Random visual localization error for the measurements from [Figure 23](#).

3.3.3 Conclusion and Future work

The parking module has proven to be robust and flexible. It allowed for performing press demonstrations with different types of parking maneuvers at different indoor and outdoor parking lots. The module is well tuned to both the actuation and perception capabilities of the V-Charge system and provides reproducible and very accurate results.

We see three main areas that can be investigated in future projects:

1. Capability to react to general traffic and changing perception data during the parking in/out maneuvers.
2. Capability to choose and perform the most efficient parking maneuver also at atypical parking spots, like those adjacent to curves or crossroads. This also includes the capability to decide - based on spot topology and the occupancy state of the neighbouring spots - whether a forward or backward parking maneuver should be performed for any given spot.
3. High density parking. This requires not only an improvement in the accuracy of the perception data but also efficient algorithms that can handle a more realistic - but also more complex - vehicle model.

We at VW plan to pursue items 1 and 2 in our future research projects. Item 3 is very relevant for automated valet parking solutions designed for exclusive (i.e. robot-only) environments.

4 3D Multilevel Operation

The software developed in this workpackage 4 has not been especially designed to support 3D environments. At certain points, 2D assumptions have been implicitly placed for the sake of simplicity, especially where associations between the current vehicle pose and the position within the road graph network have to be established. These projections assume a planar environment and might fail due to resulting ambiguities when neglecting the z -coordinate.

With some special care during the set up of the mission, avoiding these ambiguities, however, it has been demonstrated that the vehicle is fully capable of multilevel operation. During tests performed in the Schiphol Excellence Parking garage in Amsterdam, the vehicle successfully completed a mission that started in the 2nd level of the parking garage and ended at a parking spot in the 3rd level. Hereby, the vehicle had to pass a ramp connecting the two levels (see Figure 25). The mission overview is depicted in Figure 26.



Figure 25: Ramp at the Schiphol Excellence Parking garage in Amsterdam, connecting levels 2 and 3.

The picture shows the position of the vehicle at timestamp 15:37:14h in the overhead plot depicted in Figure 26. The mission started at 15:36:09h on parking level 2 and ended in the final parking spot on parking level 3 at timestamp 15:39:23h. The ramp connecting the two parking levels has been passed around timestamps 15:37:10h to 15:37:30h.

As the topology of the environment is encoded into the semantic road graph layer and correctly propagated through the planning layers, ambiguities in the middle of the path such as at position $[x, y] = [41, -5]$ (see Figure 26) do not pose a problem. A video of the run is attached to this deliverable in *V-Charge_multilevel_navigation.mp4*.

However, in general the road graph server and the global parking planner have been designed for a 2D parking area with one parking level. So far, the road graph server and the global planner use the 2D euclidean distance to connect parking spots with the closest lane and then search for the shortest global route from the current vehicle pose to the target pose. Each parking spot is connected to the closest lane of the road graph and the lanes are connected to each other.

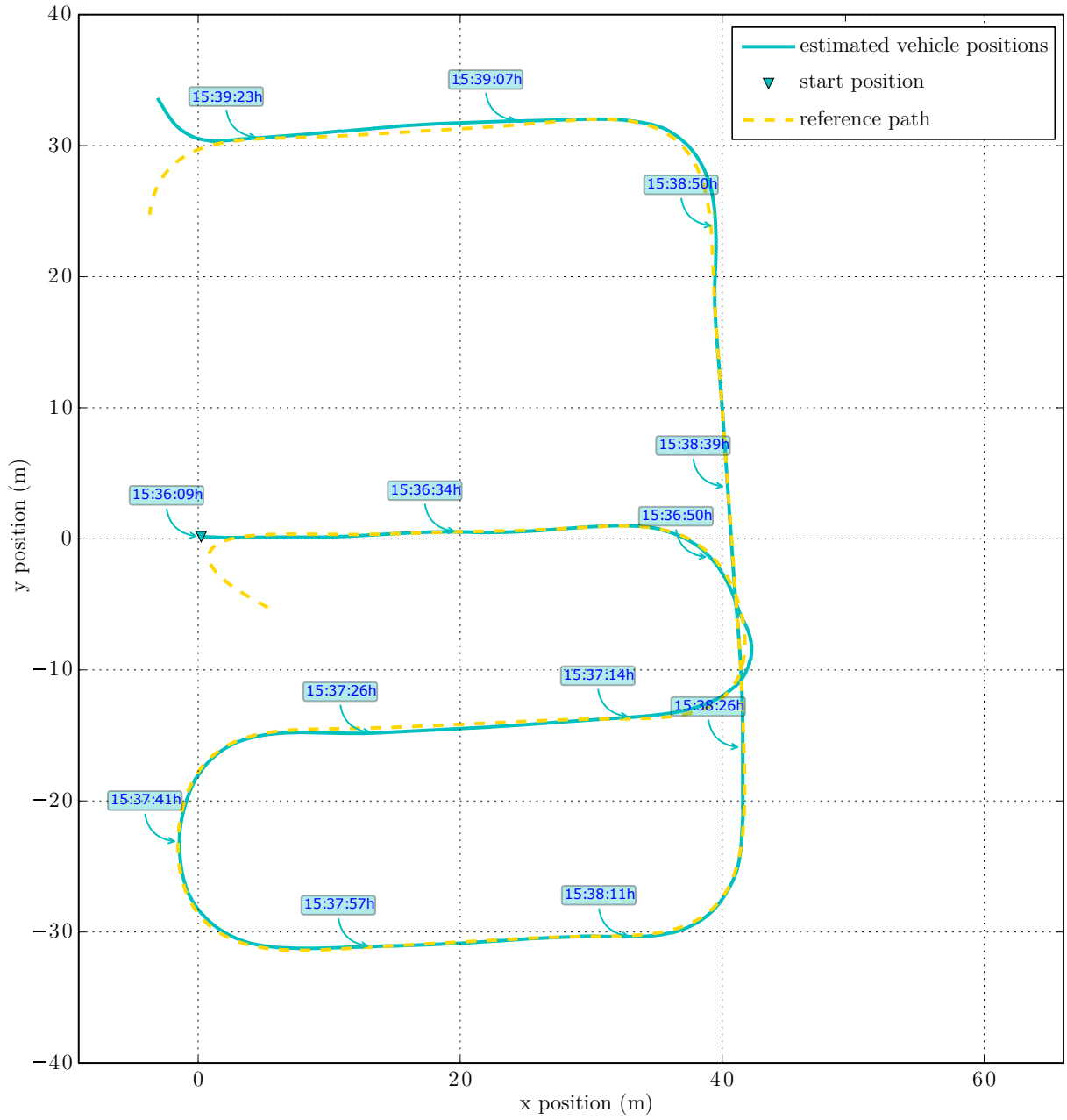


Figure 26: Overview of the mission involving two levels of the Schiphol Excellence Parking garage in Amsterdam. The cyan line depicts the vehicle pose estimations obtained from visual localization, and the yellow dotted line illustrates the global reference path computed by the global planner. Cyan tooltips show timestamped vehicle poses along the path.

This connection approach leads to false results when multiple lanes overlap. In general, there exist two approaches to enable the use of the global planner in arbitrary parking lot environments with multiple levels:

Topological connection: Instead of connecting the closest lanes and parking spots with each other, a topological connection approach could be used that correctly reflects the topology of the underlying road network. With this approach, each lane could be correctly connected over multiple levels. Also, multiple corresponding lanes to one single parking spot could be allowed, e.g. to park out in a different direction.

3D euclidean connection: In this approach, parking spots and lanes are connected to their closest neighbors by considering also the height/level-information of the z-coordinate. This means, each parking spot and all supporting points of all lanes get a third value for the z-coordinate representing the level of the parking lot environment. As described above, such an approach is not yet implemented by the road graph server and the global planner.

5 Conclusion

This deliverable reported on the final integration status and progress related to workpackage 4 as well as on the evaluation of the V-Charge planning modules in extensive tests. Additionally, important shortcomings were highlighted that have been identified during the course of the project and which are worth being addressed in future projects.

Both research and integration were successful throughout all parts of the project. All necessary components for online navigation in parking lots have been implemented, integrated and evaluated. The developed navigation modules supported multiple successful demonstrations of the V-Charge concept during the final project presentations.

References

- [Galceran et al., 2015] Galceran, E., Eustice, R. M., and Olson, E. (2015). Toward Integrated Motion Planning and Control using Potential Fields and Torque-based Steering Actuation for Autonomous Driving. In *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*.
- [Mühlfellner et al., 2015] Mühlfellner, P., Bürki, M., Bosse, M., Derendarz, W., Philippsen, R., and Furgale, P. (2015). Summary maps for lifelong visual localization. *Journal of Field Robotics*.
- [Reeds and Shepp, 1990] Reeds, J. and Shepp, L. (1990). Optimal paths for a car that goes both forwards and backwards. *Pacific Journal of Mathematics*, 145(2):367–393.
- [Schwesinger et al., 2013] Schwesinger, U., Rufli, M., Furgale, P., and Siegwart, R. (2013). A Sampling-Based Partial Motion Planning Framework for System-Compliant Navigation along a Reference Path. In *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*.
- [Schwesinger et al., 2015a] Schwesinger, U., Siegwart, R., and Furgale, P. (2015a). Fast Collision Detection Through Bounding Volume Hierarchies in Workspace-Time Space for Sampling-Based Motion Planners. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*.
- [Schwesinger et al., 2015b] Schwesinger, U., Versari, P., Broggi, A., and Siegwart, R. (2015b). Vision-only fully automated driving in dynamic mixed-traffic scenarios. *it - Information Technology*, 57(4):231–242.

A Vision-Only Fully Automated Driving in Dynamic Mixed-Traffic Scenarios

Authors: U. Schwesinger, P. Versari, A. Broggi, R. Siegwart

published in [[Schwesinger et al., 2015b](#)]: “it - Information Technology”, volume 57, issue 4 (special issue on autonomous driving), pages 231–242, ISSN (Online): 2196-7032, ISSN (Print): 1611-2776, DOI: [10.1515/itit-2015-0005](#), July 2015

The publications attached to the original deliverable were removed from the public version at hand. Please refer to the publication section on the official V-Charge webpage www.v-charge.eu.

B Fast Collision Detection through Bounding Volume Hierarchies in Workspace-time Space for Sampling-based Motion Planners

Authors: U. Schwesinger, R. Siegwart, P. Furgale

published in [Schwesinger et al., 2015a]: Proceedings of the 2015 IEEE International Conference on Intelligent Robots and Systems (ICRA), pages 63–68, DOI: [10.1109/ICRA.2015.7138981](https://doi.org/10.1109/ICRA.2015.7138981), May 2015

The publications attached to the original deliverable were removed from the public version at hand. Please refer to the publication section on the official V-Charge webpage www.v-charge.eu.