Trajectory Planning for Cooperative Autonomous Valet Parking

Reza Dariani
German Aerospace Center (DLR)
25 PUBLICATIONS **38** CITATIONS
SEE PROFILE

Julian Schindler
German Aerospace Center (DLR)
68 PUBLICATIONS **307** CITATIONS
SEE PROFILE

Trajectory Planning for Cooperative Autonomous Valet Parking

Dr.-Ing. Reza Dariani [1], Thomas Lobig [1], Jan Lauermann[1], Jonas Rieck[1], Julian Schindler [1]

[1] Institute of Transportation Systems, German Aerospace Center (DLR), Germany, reza.dariani@dlr.de, 0531-295-3436

Abstract

This paper presents the concepts and implementation of an automated valet parking algorithm with major focus on the trajectory planner and its cooperation with a parking management system. Trajectory planner approaches for parking and autonomous driving are explained and validated with simulation and experimental tests.

Keywords: Trajectory planning, autonomous driving, valet parking, parking management

1. Introduction

Modern cars are equipped with high number of driver assistance systems to increase the safety, comfort and reduce driver stress level [1]. A parking assistance system or **A**utomatic **P**arking **A**ssist **S**ystem (APAS) has been implemented for the first time in 2003 by Toyota and now exists in some cars in which the system maps the environment and detects an existing accessible parking place and parks the car into. The focus of this paper is to go one step further and develop an automated valet parking system. The general scenario of an automated valet parking system is explained in . As it is shown in panel (a), the driver transitions control to the vehicle at the drop off position, e.g. at an airport, train station, etc.. The vehicle then navigates to the parking area and drives fully automated to a free parking spot and park there (b). A free parking spot can either be directly found, using the vehicle's sensors, or received from a parking management system, e.g. via V2I communication. When the driver wants the vehicle returned, as shown in (c), he can communicate this to the vehicle via a HMI, e.g. on a smartphone. Subsequently the vehicle leaves parking spot out and drives fully automated back to a designated pick up position (d). During fully automated driving, the vehicle drives safely, regards obstacles such as other vehicles or pedestrians.
The Trajectory planner for valet parking is divided into two parts, the parking planner and the autonomous driving planner. Trajectory planners for parking are mostly based on geometric curves [2] which mostly deal with standard situations like parallel parking and perpendicular parking or they are graph-based [3]. Graph-based approaches have the possibility of solving any parking situation but they may suffer from large number of states in the graph.

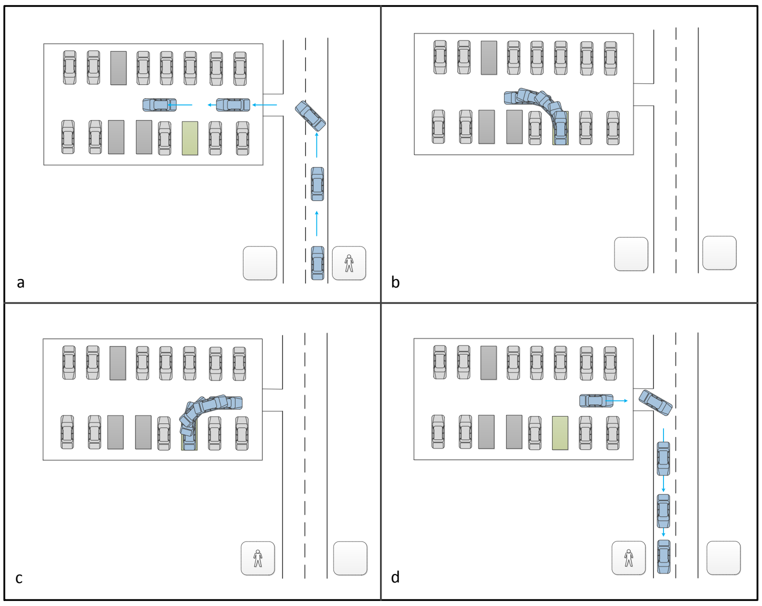

Figure 1: Automated valet parking scenario

As this paper focuses on standard parking, the geometric curves, has been chosen. Based on the vehicle position, the parking spot position and vehicle steering abilities, a set of turning radii is calculated and one which is collision free is chosen. This turning radius then is converted to vehicle inputs in real-time, employing a single-track vehicle model.
For the autonomous driving part, unlike classical approaches such as rules and maneuver based approaches [4], an optimal control based approach is used. In this approach an optimization problem is formulated and a **S**equential **Q**uadratic **P**rogramming (SQP) based solver using the quasi-newton method solves the optimization problem.
The functionality of these planners is validated using simulations and physical driving experiments.

2. Trajectory planner

As mentioned in the introduction, a geometric approach is used to plan a trajectory for parking and an optimal control based approach is used to plan a trajectory for autonomous driving mode. Both planners are explained below.

2.1 Parking planner

Based on the approach presented in [5], by considering the vehicle possible kinematic turning radius, parking trajectory is generated. In this work we focus only on a perpendicular parking scenario, , but the approach works for parallel parking as well, for more details see [5].

In case of perpendicular parking, front and side collisions as possible collision points are considered, . Turning to the parking lot may result in side collision with the car beside the parking spot (side of the vehicle and a parked car). Another collision could occur with the front end of the vehicle with vehicles parked in front or any barrier. To avoid these collisions the turn maneuver must be done with a safe turning radius. To satisfy this situation, a forward trajectory from the desired parking spot is planned to the vehicle's current position. The boundary of the turning radius for the first collision R_{fc} and the boundary of the turning radius for the second collision R_{sc} will be derived as functions of the step forward distance s, .
The boundary value of turning radii R_{fc} and R_{sc} for the vehicle's cg (center of gravity) can be simply derived including the parameter of the step forward distance, s , equation (1) and (2) respectively.

$$R_{fc} \geq \sqrt{\left(\frac{(s-b_2+k_r)^2+{b_1}^2+b_1.W}{2b_1}\right)^2+{L_r}^2} \tag{1}$$

$$\mathrm{R_{sc}} \leq \sqrt{\left(\sqrt{(\mathrm{s}-\mathrm{b_3})(\mathrm{s}-\mathrm{b_3}-2\mathrm{K_f}-2\mathrm{L})}-\frac{\mathrm{W}}{2}\right)^2+{L_r}^2} \tag{2}$$

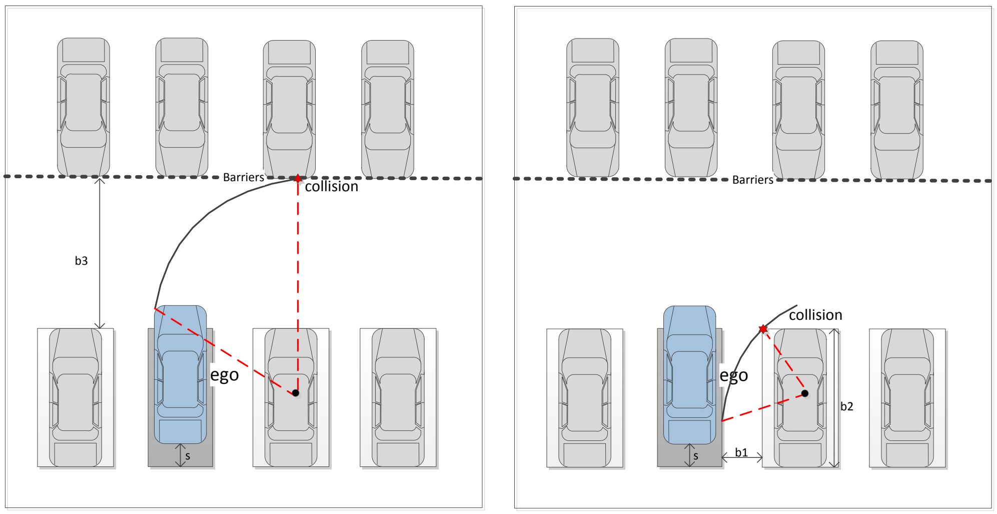

Figure 2: Two possible collisions. Left: Front collision. Right: Side collision

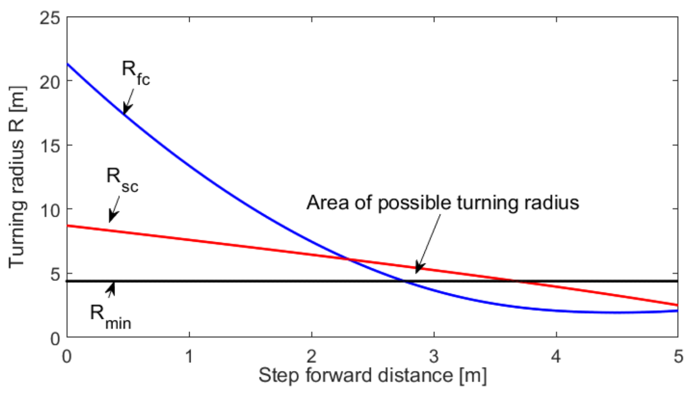

Figure 3: Turning radius and area of possible turning radius R during perpendicular parking

Where $L = L_r + L_f$ is the wheel base length, K_f is the distance between the corner of the car and the front wheel, W is width of the car, b_1 is the distance to the next parking space, b_2 is the length of the obstacle to the side and b_3 is the distance to the front obstacle.
The area of possible turning radius R is shown in as a function of step forward distance s.
Based on vehicle position and the collision free geometric curve, as result of collision free possible turning radii, a set of discrete points $\{x_p, y_{p,}\psi_p\}$ which connect the parking spot to the vehicle position is generated and by using Schoenberg and Reinsch smoothing spline approach [6] it is converted in third order splines. An optimization module containing a single track vehicle model has been used in order to convert the parking trajectory (third order splines) to vehicle actuator values. The optimization problem has the following objective function, equation (3), in which the goal is to find the vehicle actuator values which minimize the deviation of vehicle's travelled trajectory, equation (4), to the reference planned trajectory. $\mathcal{W}$ in (3) is a diagonal matrix containing weighting coefficients of each component.

$$\min J(\underline{x}) = \int_{t_0}^{t_f} \mathcal{W}\left(\underline{X} - \underline{X}_{ref}\right)^2 dt \tag{3}$$

$$\underline{X} = \{x, y, \psi\} \tag{4}$$

2.2 Autonomous driving planner

The autonomous driving trajectory planner consists of different components such as the non-linear optimal control problem as core component with its objective function defining driving behaviour; the vehicle model and the initial solution planner. A "Tactical decision" module is designed on top of the planner which defines the driving task for the planner and "Trajectory Tracking" module. This module is called "High-level controller" and it guarantees that vehicle follows the planned trajectory, see Figure 7.

Initial solution:

Optimization method used in SQP is quasi-newton based which requires a starting point also called initial solution. As the road geometry is known, planning an initial solution can be done based on the information delivered via digital map. Therefore without considering any obstacle at this stage, the shortest path is equal to driving at the center line of the road and it can be extracted directly from the digital map as a set of discrete points $(\mathrm{x_{sp}}, \mathrm{y_{sp}})$. Based on the shortest path and a kinematic single-track model, control values as initial solution can be calculated.

Optimal control problem:

In general the nonlinear optimization problem is defined as (5):

$$\min J(\underline{x}, \underline{u}) \tag{5}$$

with differential equation modelling the vehicles dynamics (6) and nonlinear constraints (7)

$$\underline{\dot{x}} = f(\underline{x}, \underline{u}) \tag{6}$$

$$\underline{g}_l \leq g(\underline{x}, \underline{u}) \leq \underline{g}_u \tag{7}$$

as well as states (8) and inputs boundaries (9)

$$\underline{x}_l \leq \underline{x} \leq \underline{x}_u \tag{8}$$

$$\underline{u}_l \leq \underline{u} \leq \underline{u}_u \tag{9}$$

The non-linearity and high length of the planning course make the optimal control problem numerically difficult to solve and also it requires high computational time. A possibility to deal with this problem is using Moving-Horizon approach *MHA* [7]. In this approach, the global optimization problem covering the complete driving task is portioned into several local optimal sub-problems of τ second, or planning horizon, which are comparatively easier to solve. The local optimal control problem structure is similar to the global problem with the difference that not the whole driving course is considered. Also in real driving scenario, the driver has limited information about road and knows only about the road ahead. The moving-horizon approach also updates the optimal control problem by saving the solution for a part of problem, named increment as a portion of horizon ξ, and it is used as the starting point for the next optimal sub-problem, see .

Vehicle model:

To describe vehicle dynamics, the single track model [8] is used. The vehicle is regarded as a rigid body moving in the xy-plane and combines both wheels per axle into one. In the vehicle model roll and pitch angles are neglected and the tire dynamics are approximated by linear tire characteristic with saturation [9]. The vehicle model (6) has the following state vector $\underline{x}$ (10) and control vector $\underline{u}$ (11).

$$\underline{x} = [x\ y\ \psi\ \dot{\psi}\ v\ \beta\ \delta\ \dot{\delta}] \tag{10}$$

$$\underline{u} = [\ddot{\delta}\ \ F_x] \tag{11}$$

The states variables are vehicle position in global coordinates $[x, y]$, vehicle yaw angle ψ and yaw rate $\dot{\psi}$, vehicle velocity v, vehicle chassis sideslip angle β, steering angle δ and steering rate $\dot{\delta}$. The control variables are steering angle acceleration $\ddot{\delta}$ to guarantee that the vehicle applied steering angle δ is smooth (two times continuous differentiable) and longitudinal force F_x**.** For further details about the vehicle models see [10]. The systems differential equation is discretized by applying Runge-Kutta integration of fourth order as numerical integrator with step size of Δt and planning horizon of $\tau = N.\Delta t$, where N is the number of integration step, see .

Objective function:

The desired driving behaviour is the result of objective function definition of the optimal control problem. Therefore the planned trajectory, as result of the defined objective function, must be collision free and comfortable for vehicle users. Objective function can be written as (12)

$$J(\underline{x}, \underline{u}) = J_{\mathcal{L}}(\underline{x}, \underline{u}) \tag{12}$$

Index $\mathcal{L}$ stands for Lagrange term, equation (13) which is an additional state inside **O**rdinary **D**ifferential **E**quation **ODE** of vehicle model (6). Steering rate $\dot{\delta}$ and steering acceleration $\ddot{\delta}$ are inside the objective function to make the steering behavior smooth and avoid uncomfortable steering wheel impulse. Δv is the difference between desired speed and vehicle current speed. Δd is the vehicle distance to the center line. $\ddot{\mathcal{X}}$ and $\dddot{\mathcal{X}}$ are acceleration and jerk in the transverse and longitudinal direction as comfort parameter. The last two terms will not prevent rapid change of direction therefore $\dot{\psi}$ is introduced to attenuates high yaw rates. And $\mathcal{W}$ is a diagonal matrix containing weighting coefficients of each component.

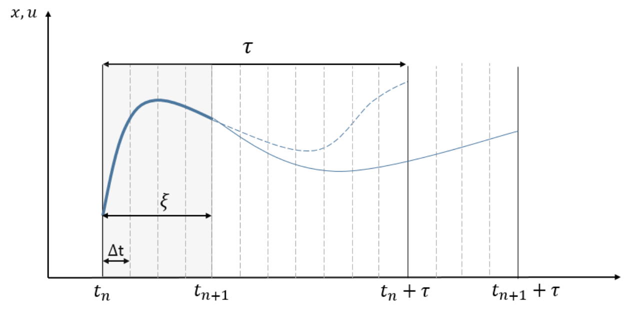

Figure 4: Moving horizon approach

$$J_{\mathcal{L}}(\underline{x},\underline{u}) = \mathcal{W}\int_{t_n}^{t_{n+\tau}} \mathcal{L}(\dot{\delta},\ddot{\delta},\Delta v,\Delta d,\ddot{\mathcal{X}},\dddot{\mathcal{X}},\dot{\psi})dt \tag{13}$$

Tactical decision:

Minimizing centrifugal acceleration, keeping safe distance to the vehicle in ahead and etc. can be formulated inside the objective function of the optimal control problem. Optimizing the above mentioned parameters in *OCP* increases the complexity of the optimization problem and calculation time. Tactical decision defines a task for trajectory planner such as driving with a given velocity or driving in a given lane module by analyzing the road geometry and predicting the future trajectory of the relevant obstacles and their intention.

Trajectory tracking:

Despite low velocity, drive in parking areas and park in a parking spot, with possible dynamic and static obstacles such as parked cars and pedestrians require precise trajectory tracking. The trajectory tracking, also called “high-level controller” must minimize the deviation between the planned trajectory and the vehicle driven trajectory and rapidly reacts to the disturbances.
To better control in case of disturbances in the system a control reserve is considered and the full kinematic vehicle ability (ex. steering angle) is not used by the trajectory planner, see (9). As the driving velocity is low, the vehicle model can be approximated linearly.
Equation (14) and (15) present the lateral controller and longitudinal controller respectively. In (14) the lateral error with e_y, heading error with e_Ψ and planned steering angle with δ_{tp} are shown. In (15) the longitudinal error with e_x, the velocity error with e_v and planned acceleration with a_{tp} are shown.

$$\delta_u = \delta_{tp} - K_y e_y + K_\Psi e_\Psi \tag{14}$$

$$a_u = a_{tp} - K_x e_x - K_v e_v \tag{15}$$

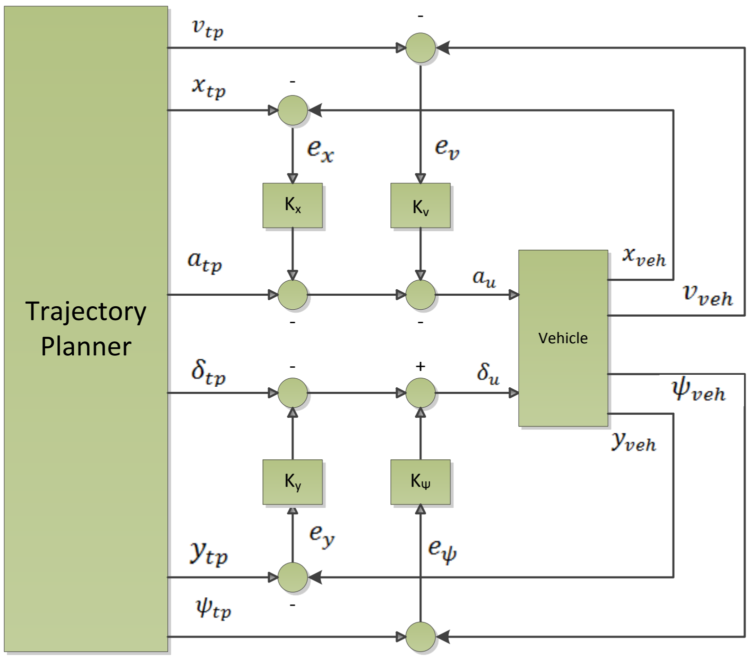

Figure 5: Trajectory tracking concept

3. Automation Software Stack

The presented trajectory planner is one module of the full automation stack used for simulation and physical driving experiments, Figure 6. This stack was developed at the DLR's Institute for Transportation Systems and is called Cooperative Safe Automation (CSA).
The main parts are represented in different libraries:

- ENV – classes and functions for environment representation
- VIEW – different views on the environment data for different purposes
- FUN – a library of classes and functions for autonomous driving functions
- MAD – a library of mathematical functions and representations used by the other libraries
- Ifmiddleware – an interface specific to an IPC middleware, to decouple the automation framework from the middleware in use.

The IPC middleware used is called Dominion [11], also developed at DLR is currently propriety software. Figure 7 presents the automated valet parking vehicle automation concept.
The logical function ordering into different modules, called apps in Dominion is as follows:

- Function Manager – coordinating communication of the car and the infrastructure and higher level decision making
- Map Provider – loading map data (OpenDrive format) into the environment representation
- Trajectory planner – separate planner for normal road driving and parking
- Highlevel Controller – a controller to have the car follow the planned trajectory

The planning method can be described by an easy meta planning, based on user intent, i.e. the command to park at a specific parking slot. The Function Manager, Figure 8, employs a simple state machine with a stack of two layers for goal management. There are three separate trajectory planners, one for normal on road autonomous driving (AD) and two different states of parking planer, one to park in a parking spot and one to leave the parking spot. If the desired goal (e.g. go to pick up) is not directly satisfied in the current state (e.g. parking in parking spot) then it is saved as "long-term goal" and an intermediate "current goal" (e.g. park out of parking spot) is planned, which brings the car closer to the long-term goal.

4. Parking Management

In the evaluation of the parking planner the basic use case was the following: a user with a smartphone issues the parking of a car using an app on the smartphone and later recalls the car to continue the ride. The first part is called "drop-off" scenario, where the car is dropped off to park autonomously. The second part is called "pick-up" scenario, when the car is picked up again by the user. While the car has the necessary information to move on the parking lot, in these scenarios we assume that it has no information about available parking spots and no intrinsic behavior to choose when to park and when to go back to pick up.

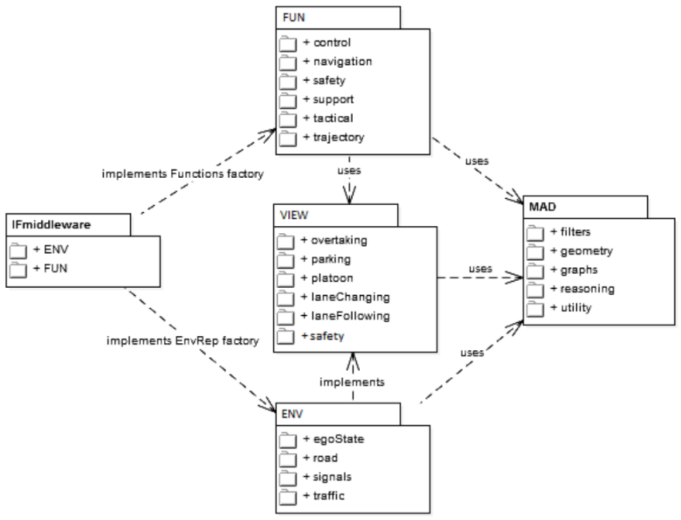

Figure 6: DLR automation software stack

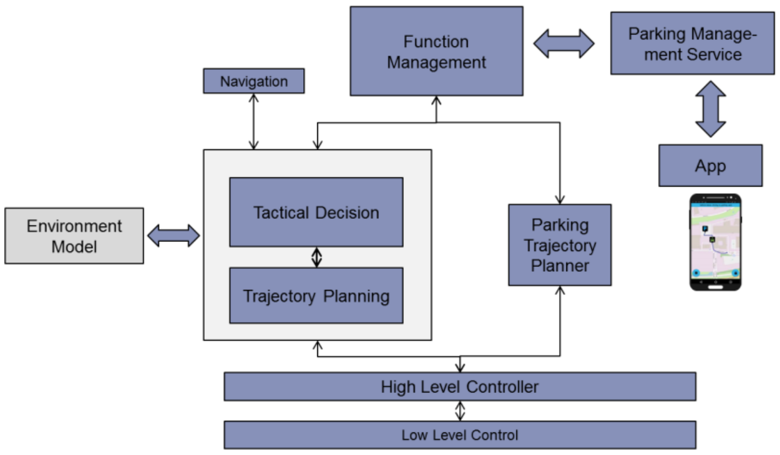

Figure 7: AVP vehicle automation architecture

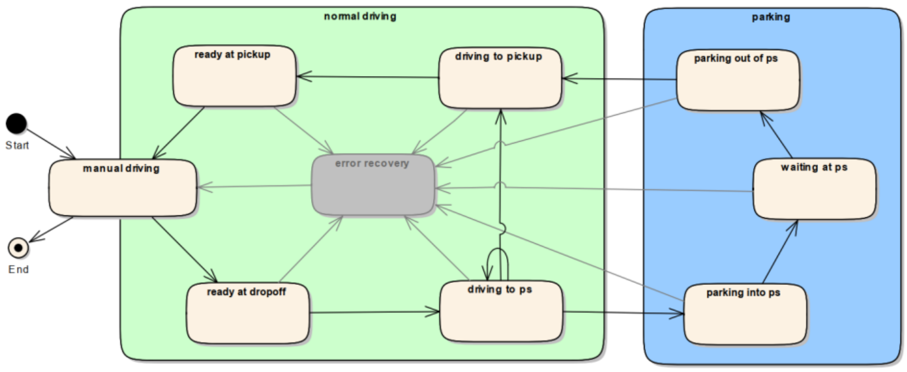

Figure 8: Function manager state machine

Therefore a backend parking management service was used. This service manages parking slot availability and brokers commands issued by the smartphone app to the respective automated car. A specific parking command is sent by the service containing all data that might be of interest concerning the given task: Parking slot dimensions and position, a suggested route to take and some more meta data. This information is used by the "Function Manager" module to control the scenario on an abstract level. While the car is using the parking lot and thus the parking service, it periodically sends status information to the parking system. Thus this system is able to manage parking spot availability, obstacles and congestion. This is however outside the scope of this paper.

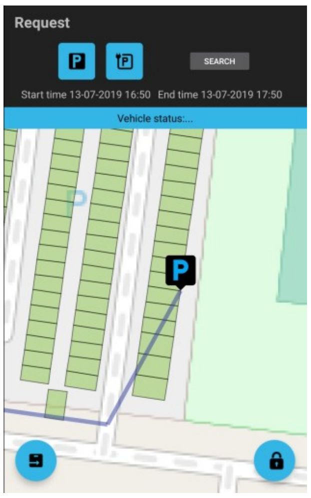

Figure 9: A screen shot of parking management mobile application

5. Simulation and experiment results

5.1 Demonstrator vehicle

For demonstrating the described scenario, DLR runs demonstrator vehicles with the ability to drive highly automated. In this case the vehicle named “FASCarE” was used.
The basic car is a VW eGolf and was modified to accept motion commands from the mentioned high-level-controller, see Figure 10. Therefore an interface was installed which accepts inputs for longitudinal acceleration, deceleration and steering angle. The installation utilizes the serial abilities of the stock car by using the **A**daptive **C**ruise **C**ontrol (ACC) control unit for longitudinal commands and the parking assistant for lateral commands. Therefore no additional actuator had to be installed to control the movement of the steering wheel for example.
For closing the loop of controlling the cars movement a high precision GNSS location system is installed in the car. The GNSS System uses differential GPS together with a precise inertial measurement unit (Novatel IMU). The accurate vehicle position is sent back to the DLR middleware and is used by the planner and the controller. The DLR middleware (Dominion) runs on the car-PCs in the trunk as well as the planning and controlling algorithms, which runs inside the DLR middleware. The interface to the car commands is done with a dSPACE MicroAutoBox (dSPACE MABXII), which translates the commands from the middleware to the car internal CAN protocol. On this device safety checks are executed prior to setting the car in motion. It is also ensured that the driver is present and that the automation activation is done by the safety driver.
For perceiving the environmental situation, the car is equipped with laser scanner. With the gathered data from the laser scanner, objects within the planned trajectory can be recognized and collisions can be avoided.

5.2 Simulation and experiment results

Figure 11 illustrates the simulation results of “Drop-off” scenario simulated in Dominion. At (a) vehicle is at drop-off position and the driver gives the control to the vehicle. Then the vehicle receives free-parking spot information from parking management and drives autonomously till vicinity of parking spot (b). At this level first a route in planned from vehicle current position to the vicinity of the parking spot and the center line of this route is converted to the initial solution of the optimal control planner. Then based on a given objective function and the task defined by tactical decision a real-time trajectory is planned. When the vehicle reaches the vicinity of the parking spot, the function management activates the parking planner and as shown a smooth trajectory to park the vehicle is planned (c). To plan such trajectory, the parking spot information and vehicle kinematic steering abilities are used. The planned trajectory is converted to the vehicle actuators by using a real-time algorithm containing the vehicle model. Then the vehicle drives backward in order to park in the given parking spot (d).
Figure 12 illustrates the same scenario with FASCarE done in the automotive campus in Helmond the Netherlands. The scenario order is like simulation as follows. At (a) the vehicle is at drop-off position and driver gives the control to the vehicle. Then vehicle drives fully autonomous to the vicinity of the desired parking spot which is shown with traffic cones (due to the safety measures traffic cones are used instead of parked vehicles) (b). Then function management activates the parking planner and after planning a parking trajectory the vehicle drives to parking spot (c) and after reaching the desired position inside the parking spot, the vehicle stops (d).

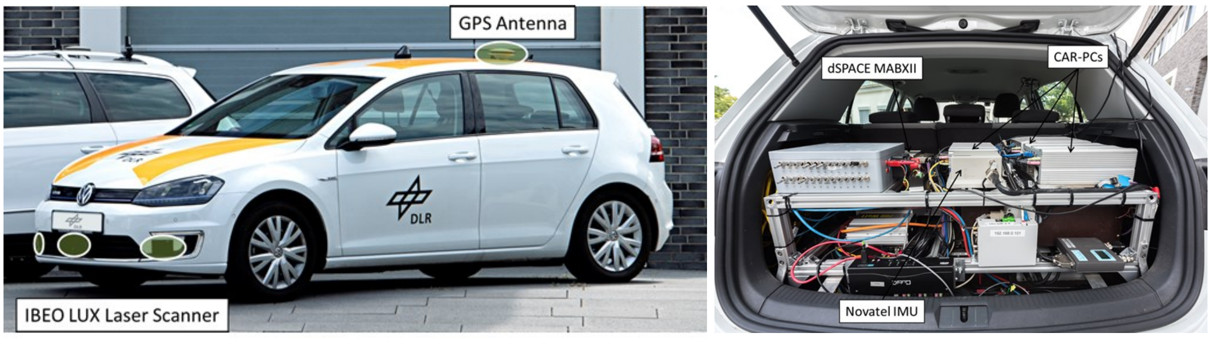

Figure 10: Left: Sensors of FASCarE. Right: Trunk installments of FASCarE

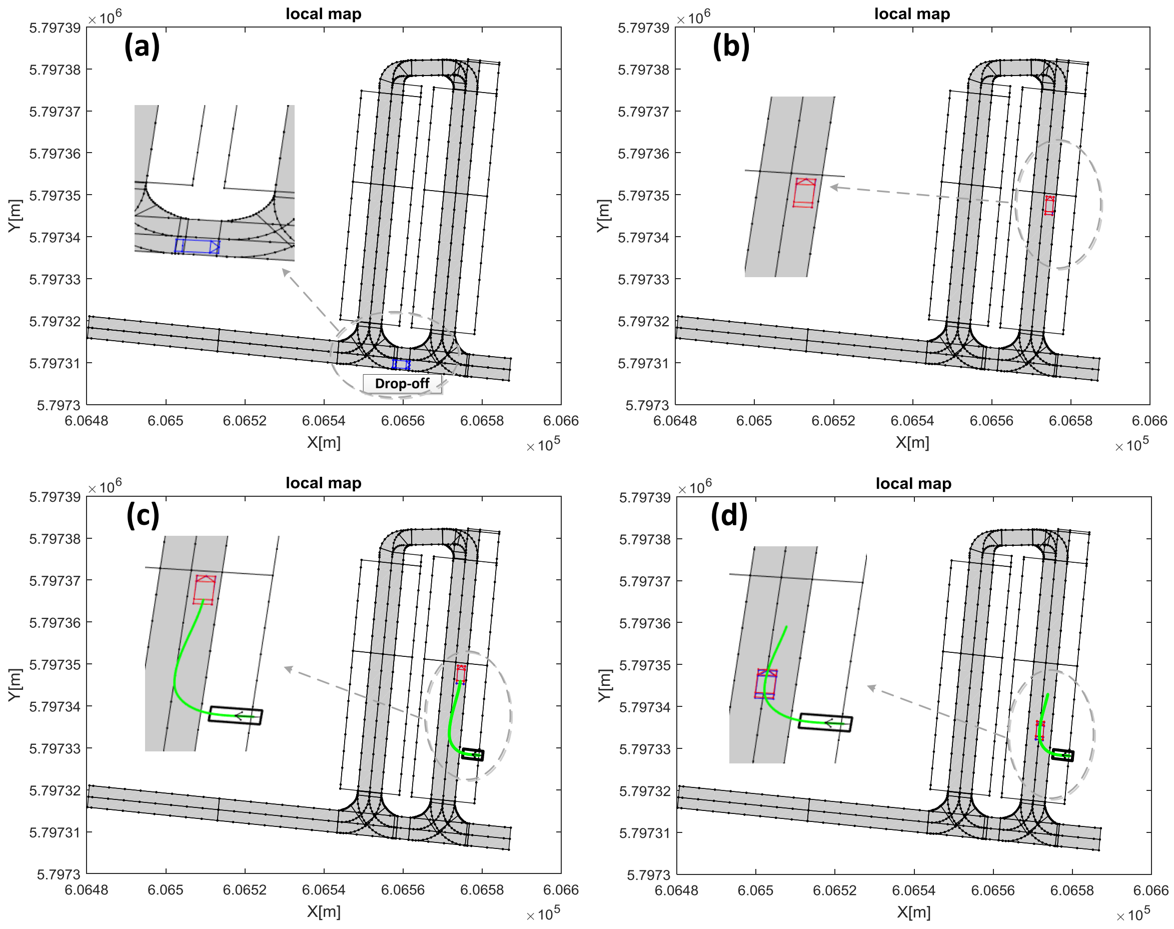

Figure 11: Drop-off scenario in simulation

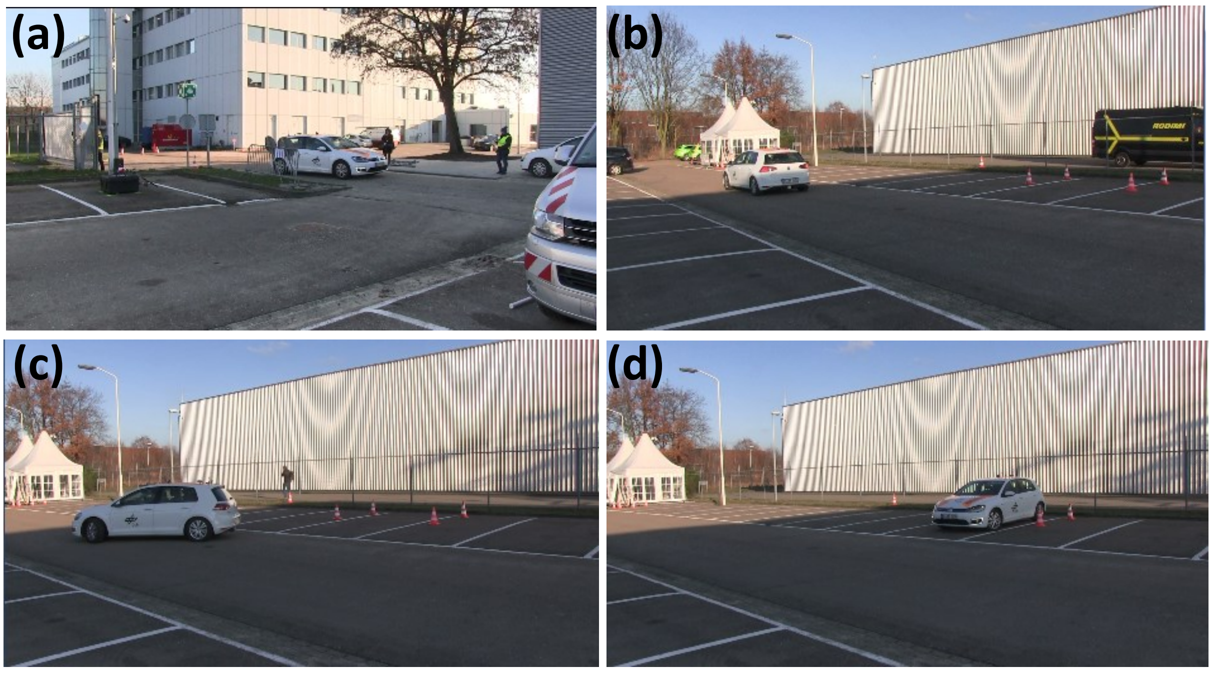

Figure 12: Drop-off scenario with FASCarE

6. Conclusion

In this paper, the DLR vehicle automation concept for automated valet parking by focusing on the trajectory planning approach has been explained. Parking planner and autonomous driving planner have been explained separately. Function management state machine and parking management system have been briefly presented and the functionality of the whole system has been proven in simulation as well as driving experiments with DLR FASCarE.

7. Bibliography

[1] R. Bryan, B. Mehler und J. F.Coughlin, „An evaluation of driver reaction to new vehicle parking assist technologies developed to reduce driver stress,“ *MIT University Transportation Centre,* pp. 1-26, 2010.

[2] H. Vorobieva, N. Minoiu-Enache, S. Glaser und S. Mammad, „Geometric continuous-curvature path planning for automated parallel parking,“ *Intern. Conf. on Networking, Sensing and Control (ICNSC),* pp. 418-423, 2013.

[3] J. Ziegler, P. Bender, T. Dang und C. Stiller, „Trajectory planning for Bertha- a local, contiuous method,“ *IEEE Intelligent Vehicles Symposium Proceedings,* pp. 450-457, 2014.

[4] A. Kelly und A. Stentz, „Rough terrain autonomous mobility-Part 1: A theoretical analysis of requirements,“ *Autonomous Robots 5,* pp. 129-161, 1998.

[5] J. Moon, I. Bae, J.-g. Cha und S. Kim, „A trajectory planning method based on forward path generation and backward tracking algorithm for automated parking systems,“ *2014 IEEE 17th International Conference on Intelligent Transportation Systems (ITSC),* 8-11 October 2014.

[6] C. d. Boor, A practical guide to splines, Springer-Verlag Newyork.

[7] M. Gerdts, „A moving horizon technique for the simulation of automobile test drives,“ *Journal of applied mathematics and mechanics,* pp. 147-162, 25 February 2003.

[8] R. Mayr, „Verfahren zur Bahnfolgeregelung für ein autonom geführtes Fahrzeug,“ University of Dortmund, Dortmund, 1991.

[9] H. B. Pacejka, Tire and vehicle dynamics, SAE International and Butterworth Heinemann, 2012.

[10] R. Dariani, „Hierarchical concept of optimization based path planning for autonomous driving,“ Otto-von-Guericke University of Magdeburg, Magdeburg, 2016.

[11] B. Hendriks, C. Harms und M. Kürschner, „Dominion- A realtime middleware for connecting functions in highly automated vehicles,“ *Internationales Verkehrswesen 71,* pp. 29-33.