Fast Optimization Based Motion Planning and Path-Tracking Control for Car Parking

Patrik Zips * **Martin Böck** * **Andreas Kugi** *

* *Automation and Control Institute, Vienna University of Technology, Gusshausstrasse 27–29, 1040 Vienna, Austria (e-mail: {zips,boeck,kugi}@acin.tuwien.ac.at)*

Abstract: This paper presents a car parking control concept for real-time application. It utilizes a two-degrees-of-freedom control scheme consisting of a feedforward and a feedback controller. The reference trajectory is constructed in two steps. First a geometric path is planned by solving a local static optimization problem, which is formulated by discretizing the path. Second a path-following problem in the form of an optimal control problem considering the physical limitations is solved. The solution of this optimal control problem yields the time parametrization of the geometric path. In order to account for model uncertainties and disturbances, a Lyapunov-based feedback controller is designed for the trajectory error system. Simulation studies show the applicability and efficiency of the proposed approach.

Keywords: Path planning, Optimization, Nonlinear control, Real-Time, Autonomous vehicles

1. INTRODUCTION

One topic of recent research in automobile industry is autonomous driving, which is especially challenging in urban environments. A special subject within this area of research is autonomous parking control, where close distances to obstacles and multiple driving direction changes have to be handled under consideration of the non-holonomic constraints of the car.

A common approach to tackle this challenge is to subdivide the task into a kinematic and a dynamic subproblem by means of the so-called Path-Velocity-Decomposition (Kant and Zucker, 1986). The kinematic subproblem can be solved geometrically, e.g., by stringing together lines and arcs (Hsieh and Özgüner, 2008; Kim et al., 2010). Other approaches calculate a holonomic path and move along it by considering the non-holonomic constraints (Laumond et al., 1994; Müller et al., 2007).

For the dynamic subproblem, henceforth referred to as path-following, different solutions are reported in the literature, see, e.g., (Nielsen et al., 2010) for a geometric approach and (Aguiar et al., 2004) for Lyapunov-based methods. Faulwasser et al. (2011) introduced an efficient, optimization based concept for differentially flat systems, which systematically accounts for the physical constraints of the car.

The stabilization of the trajectory error system is achieved by various techniques, e.g., the path deformation method (Khatib et al., 1997), Lyapunov-based controllers (Siciliano and Khatib, 2008) or linear control strategies (Hermosillo and Sekhavat, 2003).

In this paper, we present a trajectory generator consisting of a novel fast geometric path planner and an optimization based solution for the path-following problem similar to Faulwasser et al. (2011). The geometric planner consists of a series of static optimization problems, which is obtained by discretising the path. Obstacles are described by inequality constraints based on the Minkowski sum. The formulation of this optimization problem was tailored to be solved numerically in a very efficient way.

The path-following problem exploits the flatness property of the car enabling the reformulation of the system dynamics in form of a linear second order differential equation. The solution of an optimal control problem subject to these system dynamics considering maximum velocity, acceleration and steering angle speed calculates a time evolution along the geometric path.

Additionally, a trajectory tracking feedback controller is presented. To this end, the error system is transformed into a chained-form system. The control inputs are calculated using an integrator backstepping method and the stability of the closed-loop system is proven utilizing Barbalat's lemma.

The paper is organized as follows: After the problem statement in Section 2, the basic concept of the geometric path planner is presented in Section 3. Section 4 is devoted to the path-following and Section 5 to the feedback controller design. Simulation studies are carried out in Section 6.

2. PROBLEM STATEMENT

In this paper, we strive for a systematic and real-time capable control scheme for autonomous car parking. This requires the calculation of a trajectory in narrow environments. The trajectory has to consider the physical limitations of the car as well as its non-holonomic constraints. As the environment is usually dynamic the planning process has to be fast enough to be able to calculate a new trajectory in real-time if changes are detected. Additionally, an efficient feedback controller is needed to account for model errors and disturbances.

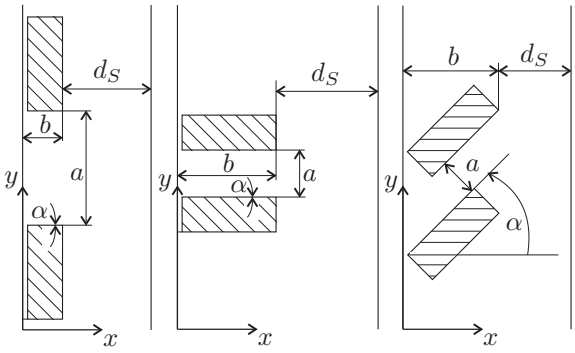

Fig. 1. Notation for parallel, garage and angle parking.

2.1 Environment

Subsequently, three common parking scenarios, namely parallel, garage and angle parking, will be considered. These scenarios and the corresponding notations are shown in Figure 1, whereby the shaded polygons represent obstacles like other cars. The lines to the left and right side of each scenario are boundaries, which shall not be violated like, e.g., the kerbstone or the lane separator of the street.

Without loss of generality we concentrate on convex polygonal obstacles. Every non-convex polygon can be subdivided into multiple convex polygons, which is usually referred to as convex decomposition (Liu et al., 2010). A non-polygonal obstacle can always be included into a slightly larger polygon.

2.2 System Dynamics

The mathematical model of the car is based on the simple kinematic model with Ackerman steering as depicted in Figure 2. The tire slip angle is neglected, which is justified at the low velocities during parking manoeuvres. Hence the car can be described by one front and one rear wheel. The motion of the car is characterised by the coordinates (x, y) of a reference point P_R, which is located at the centre of the rear axle, as well as the orientation θ of the longitudinal axis of the car. The differential equations in the state $\mathbf{q} = [x, y, \theta, v, \delta]^T$, with the velocity v and the steering angle δ of the car, read as

$$\dot{\mathbf{q}} = \begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{v} \\ \dot{\delta} \end{pmatrix} = \begin{pmatrix} v\cos(\theta) \\ v\sin(\theta) \\ \frac{v}{L}\tan(\delta) \\ a \\ \zeta \end{pmatrix}. \tag{1}$$

Here $\mathbf{u}_A = [a, \zeta]^T$ describes the control input composed of the acceleration a and the steering angle speed ζ, and L denotes the wheelbase.

3. PATH PLANNING

Considering the dynamic model (1) for the motion planning results in high computational costs. Therefore, the kinematic path planning problem is separated from the dynamic velocity planning problem by means of the well-known Path-Velocity-Decomposition (Kant and Zucker, 1986). This reduces the problem to find a feasible geometric path, which is parametrised in the parameter s. To

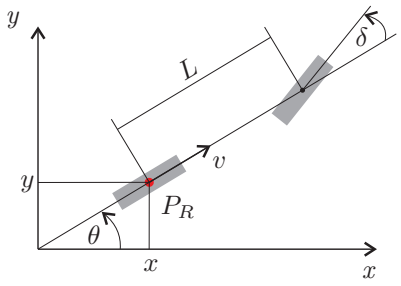

Fig. 2. Kinematic model of the car.

this end, the velocity can be expressed as $v = D\frac{\mathrm{d}s}{\mathrm{d}t}$, where $D \in \{-1, 1\}$ refers to the driving direction of the car, with $D = 1$ for velocities $v \geq 0$ and $D = -1$ for $v < 0$. Thus, the first three differential equations of (1) can be rewritten as

$$\bar{\mathbf{q}}' = \begin{pmatrix} x' \\ y' \\ \theta' \end{pmatrix} = \begin{pmatrix} D\cos(\theta) \\ D\sin(\theta) \\ Du_l \end{pmatrix}, \tag{2}$$

where $(\cdot)'$ denotes the derivative with respect to the path parameter s and $u_l = \tan(\delta)/L$ is the new control input.

To obtain a feasible path, a fast geometric path planner based on a constrained static optimization problem is used. In the following, we give a short explanation of the basic idea of the path planner. Due to page limitations, a detailed description of this algorithm is beyond the scope of this paper.

In a first step, (2) is discretised with respect to the path parameter s by means of a Runge-Kutta discretisation of second order yielding the difference equations

$$\bar{\mathbf{q}}_{i+1} = \begin{pmatrix} x_i + D\eta_i \cos\left(\theta_i + D\frac{\eta_i u_{l_i}}{2}\right) \\ y_i + D\eta_i \sin\left(\theta_i + D\frac{\eta_i u_{l_i}}{2}\right) \\ \theta_i + D\eta_i u_{l_i} \end{pmatrix} = \mathbf{f}(\bar{\mathbf{q}}_i, \mathbf{u}_i, D), \tag{3}$$

with the step length $\eta_i \in [\eta_{min}, \eta_{max}]$. The new control input $\mathbf{u} = [u_l, \eta]^T$ consists of the steering input u_l and the step length η. Note that the step length is constrained by a minimum and maximum value, η_{min} and η_{max}, respectively. The steering input u_l is limited to the maximum steering input $u_{l,m} = \tan(\delta_m)/L$, with the maximum steering angle δ_m.

The whole path can now be composed of N steps, whereby N is not fixed in advance. In each step, a constrained static optimization problem is solved to determine the control input $\mathbf{u}_i$, where the end point of the step $(i-1)$ serves as the starting point of the i^{th} step. An illustration of this procedure is shown in Figure 3. The whole path is planned in backward direction from the parking position $\bar{\mathbf{q}}_P$ to the starting position $\bar{\mathbf{q}}_S$. In this example, four iterations have already been performed leading to the positions $\bar{\mathbf{q}}_i$, $i = 1, \ldots, 4$. For the fifth iteration a suitable steering input u_{l_5} and step length η_5 have to be found leading to a position $\bar{\mathbf{q}}_5$ inside the shaded arc. The shaded arc illustrates the feasible region determined by the constraints. Note that for illustration purpose the step length is depicted much larger than at the actual planning process.

To find a suitable control input for each step i, the constrained static optimization problem

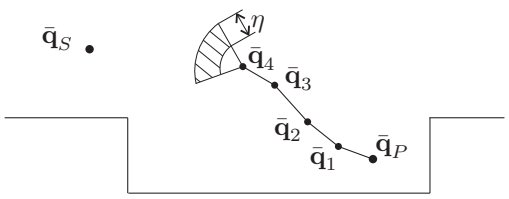

Fig. 3. Basic idea of the geometric path planner.

$$\min_{\mathbf{u}_i} \; l_{O_i}(\bar{\mathbf{q}}_{i+1}) \tag{4a}$$

$$\text{s.t. } \bar{\mathbf{q}}_{i+1} = \mathbf{f}(\bar{\mathbf{q}}_i, \mathbf{u}_i, D) \tag{4b}$$

$$\mathbf{u}_{min} \leq \mathbf{u}_i \leq \mathbf{u}_{max} \tag{4c}$$

$$\mathbf{h}_P\left(\bar{\mathbf{q}}_{i+1}\right) \leq \mathbf{0}, \tag{4d}$$

with the cost function

$$l_{O_i}(\bar{\mathbf{q}}_{i+1}) = r_\theta e_{\theta_{i+1}}^2 + \mathbf{e}_{P_{i+1}}^T \mathbf{R} \mathbf{e}_{P_{i+1}} \tag{5}$$

is defined. Thereby, $\mathbf{e}_{P_i} = [x_i - x_S, y_i - y_S]^T$ denotes the distance of the car at step i to the starting position $[x_S, y_S]^T$, and $e_{\theta_i} = \theta_i - \theta_O$ refers to the difference between the car orientation and a predefined target angle θ_O. The parameter $r_\theta > 0$ and the positive semi-definite matrix $\mathbf{R}$ serve as weighting terms in the cost function. The constraint (4b) corresponds to (3) and the control input $\mathbf{u}_i$ is constrained by $\mathbf{u}_{min} = [-u_{l,m}, \eta_{min}]^T$ and $\mathbf{u}_{max} = [u_{l,m}, \eta_{max}]^T$. For collision avoidance the boundaries of the obstacles are transformed into the inequality constraint (4d) by means of the Minkowski-sum (Lozano-Perez, 1983).

If a driving direction change should be necessary for obstacle avoidance or to reach the starting position, two heuristic rules mimicking the behaviour of a human driver are implemented. If these rules apply, a direction switching point is added at this position and the driving direction is set to $D = -D$.

A detailed description of these rules as well as the choice of the weighting parameters and the target angle θ_O is going to be published in an upcoming publication. Although this is a local planning algorithm, global convergence for specific geometric conditions can be guaranteed.

4. PATH-FOLLOWING

The path obtained by the planner described in the previous section consists of $n_{SP} + 1$ segments, where n_{SP} corresponds to the number of direction switching points. An example for $n_{SP} = 3$ path segments is illustrated in Figure 4.

Each segment is parametrised in the path parameter s. To follow this geometric path, a mapping

$$t \to s_j(t), \quad s_j(t_{0,j}) = s_{0,j}, \; s_j(T_j) = s_{E,j} \tag{6}$$

for each path segment j, with the path start and end positions $s_{0,j}$ and $s_{E,j}$, is needed. This mapping is required to consider the physical limits of the maximum velocity v_m, acceleration a_m and steering angle speed ζ_m. The geometric path planner already takes into account the limit of the steering angle δ_m.

To address this issue, we pick up an approach proposed by Faulwasser et al. (2011) concerning path-following for flat systems (Fliess et al., 1995). Hereby, the path-following

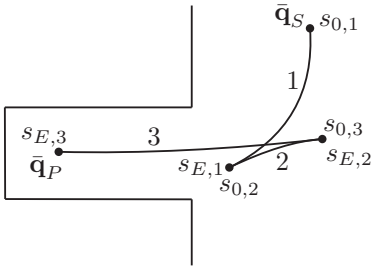

Fig. 4. Geometric path with $n_{SP} = 3$ segments.

problem is reformulated as a linear SISO system to find a timing law for the path evolution. The flatness of the car model (1) as well as the implementation of this approach are described in next subsections.

4.1 Flatness of the car

All states and control inputs of a flat system can be parametrised in terms of a flat output ξ and its time derivatives. For the model (1), a flat output is given by $\xi = [x, y]^T$. The parametrisation of the state $\mathbf{q}$ is obtained in the form

$$\theta = \arctan\left(\frac{\dot{y}}{\dot{x}}\right) \tag{7a}$$

$$v = D\sqrt{\dot{x}^2 + \dot{y}^2} \tag{7b}$$

$$\delta = \arctan\left(\frac{L}{D}\frac{\dot{x}\ddot{y} - \dot{y}\ddot{x}}{(\dot{x}^2 + \dot{y}^2)^{\frac{3}{2}}}\right). \tag{7c}$$

The control input $\mathbf{u}_A$ can be parametrised by differentiating (7b) and (7c) resulting in

$$a = D\frac{\dot{x}\ddot{x} + \dot{y}\ddot{y}}{\sqrt{\dot{x}^2 + \dot{y}^2}} \tag{8a}$$

$$\zeta = \frac{L}{D}\frac{(v\ddot{\theta} - \dot{\theta}a)}{v^2 + L^2\dot{\theta}^2}, \tag{8b}$$

where $\dot{\theta}$ and $\ddot{\theta}$ are the time derivatives of (7a). The drawback of this parametrisation is the singularity for $v = 0$, which occurs at the starting and parking position as well as at the driving direction switching points. A method to avoid this singularity is to replace derivatives with respect to the time t by derivatives with respect to the path parameter s. For a function $\sigma(s(t))$ the relations

$$\dot{\sigma} = \dot{s}\sigma' \tag{9a}$$

$$\ddot{\sigma} = \ddot{s}\sigma' + \dot{s}^2\sigma''. \tag{9b}$$

hold. Applying these relations to (7) and (8) yields the parametrisation of the states and the control inputs

$$\theta = \arctan\left(\frac{y'}{x'}\right) \tag{10a}$$

$$v = \dot{s}\sigma_1 \tag{10b}$$

$$\delta = \arctan\left(\frac{L}{D}\frac{x'y'' - y'x''}{\sigma_1^3}\right) \tag{10c}$$

$$a = \frac{\ddot{s}\sigma_1^2 + \dot{s}^2\sigma_2}{\sigma_1} \tag{10d}$$

$$\zeta = \frac{L}{D}\frac{\dot{s}(\theta''\sigma_1^2 - \theta'\sigma_2)}{\sigma_1^3\left(1 + \frac{L^2\theta'^2}{\sigma_1^2}\right)} \tag{10e}$$

with

$$\sigma_1 = D\sqrt{x'^2 + y'^2} \quad \text{and} \quad \sigma_2 = x'x'' + y'y''. \tag{11}$$

Since $\sqrt{x'^2 + y'^2} = 1$ according to (2), (11) simplifies to

$$\sigma_1 = D \quad \text{and} \quad \sigma_2 = 0, \tag{12}$$

and thus the parametrisation for the velocity and the control inputs read as

$$v = D\dot{s} \tag{13a}$$
$$a = D\ddot{s} \tag{13b}$$
$$\zeta = \frac{L}{D}\frac{\dot{s}\theta''}{1 + L^2\theta'^2}. \tag{13c}$$

4.2 Trajectory Generation

The system state $\mathbf{q}$ and the control input $\mathbf{u}_A$ are parametrised by the flat output $\xi(s) = [x(s), y(s)]^T$, the path parameter s and its time derivatives. The geometric path planner from Section 3 provides the path segments $\xi_j(s_j) = [x_j(s_j), y_j(s_j)]^T, j = 1, \ldots, n_{SP} + 1$. To obtain a suitable trajectory, a time evolution of the path parameter is needed, which ensures that the constraints of the car are not violated. This task is carried out for each path segment j separately yielding $s_j(t)$.

To this end, the evolution of the path parameter is described by the linear system (Faulwasser et al., 2011)

$$\dot{\boldsymbol{\chi}}_j = \begin{pmatrix} \chi_{2,j} \\ \kappa_j \end{pmatrix}, \tag{14}$$

with the new state $\boldsymbol{\chi}_j = [\chi_{1,j}, \chi_{2,j}]^T = [s_j, \dot{s}_j]^T$ and the control input $\kappa_j = \ddot{s}_j$. A time parametrisation of the path segment j has to fulfil the terminal conditions $\boldsymbol{\chi}_j(0) = [s_{0,j}, 0]^T$ and $\boldsymbol{\chi}_j(T_j) = [s_{E,j}, 0]^T$. A feasible control input κ_j, which considers the physical limitations of the car, is obtained as a solution of the constrained optimal control problem (OCP) with free end time T_j

$$\min_{\kappa_j(\cdot), T_j} J(\kappa_j(\cdot)) = \min_{\kappa_j(\cdot), T_j} \int_0^{T_j} 1 + r_{\chi_2}\chi_{2,j}^2 + r_\kappa \kappa_j^2 \mathrm{d}t \tag{15a}$$
$$\text{s.t. } \dot{\boldsymbol{\chi}}_j = [\chi_{2,j}, \kappa_j]^T, \qquad \boldsymbol{\chi}_j(0) = [s_{0,j}, 0]^T \tag{15b}$$
$$\boldsymbol{\chi}_j(T_j) = [s_{E,j}, 0]^T$$
$$|v(\boldsymbol{\chi}_j)| \leq v_m \tag{15c}$$
$$|a(\boldsymbol{\chi}_j)| \leq a_m \tag{15d}$$
$$|\zeta(\boldsymbol{\chi}_j)| \leq \zeta_m. \tag{15e}$$

The functions $v(\boldsymbol{\chi}_j)$, $a(\boldsymbol{\chi}_j)$ and $\zeta(\boldsymbol{\chi}_j)$ in (15c)-(15e) directly follow from (13).

The whole trajectory of the parking manoeuvre is obtained by solving the OCP (15) for each path segment $j = 1, \ldots, n_{SP} + 1$. This is numerically still more efficient than calculating the whole trajectory at once, since at the switching points the geometric path is not continuously differentiable. Note that for each segment the time can be transformed to $t_j \in [0, T_j]$ by $t_j = t - \sum_{k=1}^{j-1} T_k$.

By using the obtained time parametrisation $s_j(t)$, $j = 1, \ldots, n_{SP} + 1$, the desired state $\mathbf{q}_d(t)$ and the desired control inputs $\mathbf{u}_{A,d}(t)$ can be calculated from (10) and (13) yielding the feedforward part of the controller.

5. FEEDBACK CONTROLLER

For the feedback controller design, we introduce a reference vehicle

$$\dot{\mathbf{q}}_d = \begin{pmatrix} \dot{x}_d \\ \dot{y}_d \\ \dot{\theta}_d \\ \dot{v}_d \\ \dot{\delta}_d \end{pmatrix} = \begin{pmatrix} v_d\cos(\theta_d) \\ v_d\sin(\theta_d) \\ \frac{v_d}{L}\tan(\delta_d) \\ a_d \\ \zeta_d \end{pmatrix} = \mathbf{f}_d(\mathbf{q}_d, \mathbf{u}_{A,d}). \tag{16}$$

Let

$$\begin{pmatrix} x_e \\ y_e \\ \theta_e \end{pmatrix} = \begin{bmatrix} \cos(\theta_d) & \sin(\theta_d) & 0 \\ -\sin(\theta_d) & \cos(\theta_d) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x - x_d \\ y - y_d \\ \theta - \theta_d \end{pmatrix} \tag{17}$$

denote the tracking error of the car to the reference vehicle expressed in a Frenèt frame attached to the reference vehicle. We look for a controller that stabilizes the tracking error system (17) asymptotically. Therefore, similar to Siciliano and Khatib (2008), a new state $\mathbf{z} = [z_1, z_2, z_3, z_4, z_5]^T$ is introduced as

$$z_1 = x_e \tag{18a}$$
$$z_2 = y_e \tag{18b}$$
$$z_3 = \tan(\theta_e) \tag{18c}$$
$$z_4 = \frac{\tan(\delta) - \cos(\theta_e)\tan(\delta_d)}{L\cos^3(\theta_e)} + k_2 z_2 \tag{18d}$$
$$z_5 = v\cos(\theta_e) - v_d, \tag{18e}$$

with $k_2 > 0$. The term $k_2 z_2$ is added to be able to influence y_e, which would not be the case otherwise. Differentiating (18) yields the system dynamics in chained-form

$$\dot{z}_1 = \frac{v_d}{L}\tan(\delta_d) z_2 + z_5 \tag{19a}$$
$$\dot{z}_2 = -\frac{v_d}{L}\tan(\delta_d) z_1 + v_d z_3 + z_3 z_5 \tag{19b}$$
$$\dot{z}_3 = -k_2 v_d z_2 + v_d z_4 + z_5\left(z_4 - k_2 z_2 + (1 + z_3^2)\frac{\tan(\delta_d)}{L}\right) \tag{19c}$$
$$\dot{z}_4 = w_2 \tag{19d}$$
$$\dot{z}_5 = w_3 \tag{19e}$$

with the abbreviated control inputs

$$w_2 = k_2\dot{y}_e + \left(3\frac{\tan(\delta)}{\cos(\theta_e)} - 2\tan(\delta_d)\right)\frac{\sin(\theta_e)}{L\cos^3(\theta_e)}\dot{\theta}_e - \frac{\zeta_d}{L\cos^2(\delta_d)\cos^2(\theta_e)} + \frac{\zeta}{L\cos^2(\delta)\cos^3(\theta_e)} \tag{20a}$$
$$w_3 = a\cos(\theta_e) - v\dot{\theta}_e\sin(\theta_e) - a_d. \tag{20b}$$

The control input for the car $\mathbf{u}_A = [a, \zeta]^T$ can be obtained from (20)

$$a = \frac{1}{\cos(\theta_e)}\left(w_3 + v\dot{\theta}_e\sin(\theta_e) + a_d\right) \tag{21a}$$
$$\zeta = L\cos^2(\delta)\cos^3(\theta_e)\Bigg(w_2 - k_2\dot{y}_e - \left(3\frac{\tan(\delta)}{\cos(\theta_e)} - 2\tan(\delta_d)\right)\frac{\sin(\theta_e)}{L\cos^3(\theta_e)}\dot{\theta}_e + \frac{\zeta_d}{L\cos^2(\delta_d)\cos^2(\theta_e)}\Bigg). \tag{21b}$$

In a first step, we want to stabilize the subsystem (19a)-(19d) with the control input w_2 and the virtual control input $z_5 = w_1$. In Siciliano and Khatib (2008) it is shown that the feedback law of the form

$$w_1 = -k_1 v_d^2 \left(z_1 + \frac{z_3}{k_2} \left(z_4 + \frac{1+z_3^2}{L} \tan(\delta_d) \right) \right) \quad (22a)$$

$$w_2 = -k_4 z_4 v_d^2 - k_3 v_d z_3, \quad k_1 > 0, k_3 > 0, k_4 > 0 \quad (22b)$$

renders the closed-loop system stable in the sense of Lyapunov. For this, the Lyapunov function is chosen as

$$V_1 = \frac{1}{2} z_1^2 + \frac{1}{2} z_2^2 + \frac{1}{2k_2} z_3^2 + \frac{1}{2k_2 k_3} z_4^2 > 0 \quad (23)$$

and the time derivative of V_1 along a solution curve of (19a)-(19d) with $z_5 = w_1$ and w_2 according to (22b) takes the form

$$\begin{aligned} \dot{V}_1 =& \frac{z_4}{k_2 k_3} w_2 + \frac{v_d z_3 z_4}{k_2} \\ &+ w_1 \left(z_1 + \frac{z_3}{k_2} \left(z_4 + \frac{1+z_3^2}{L} \tan(\delta_d) \right) \right) \end{aligned} \quad (24)$$

or

$$\begin{aligned} \dot{V}_1 =& W_1 = -k_4 v_d^2 z_4^2 \\ &- k_1 v_d^2 \left(z_1 + \frac{z_3}{k_2} \left(z_4 + \frac{1+z_3^2}{L} \tan(\delta_d) \right) \right)^2 \leq 0, \end{aligned} \quad (25)$$

respectively. Furthermore, under the assumption of a uniformly continuous $v_d(t) \neq 0$ and $\delta_d(t)$ application of Barbalat's lemma yields the asymptotic result

$$\lim_{t \to \infty} z_j(t) = 0, \quad j = 1, \ldots, 4. \quad (26)$$

Since w_1 is only a virtual control input, the feedback law for the real control input w_3 is obtained by employing the integrator backstepping approach. According to the backstepping standard procedure, see (Krstic et al., 1995), an augmented Lyapunov function of the form

$$V_2 = V_1 + \frac{1}{2}(z_5 - w_1)^2 > 0 \quad (27)$$

is introduced. The time derivative along a solution curve of the tracking error system (19) with w_2 from (22b) reads as

$$\begin{aligned} \dot{V}_2 =& W_1 + (z_5 - w_1) \Bigg(w_3 - \dot{w}_1 + z_1 \\ &+ \frac{z_3}{k_2} \left(z_4 + \frac{1+z_3^2}{L} \tan(\delta_d) \right) \Bigg). \end{aligned} \quad (28)$$

It can be immediately seen that the feedback law

$$\begin{aligned} w_3 =& \dot{w}_1 - z_1 - \frac{z_3}{k_2} \left(z_4 + \frac{1+z_3^2}{L} \tan(\delta_d) \right) \\ &- k_5 (z_5 - w_1), \qquad k_5 > 0, \end{aligned} \quad (29)$$

thus results in

$$\dot{V}_2 = W_1 - k_5 (z_5 - w_1)^2 \leq 0, \quad (30)$$

guaranteeing the stability of the closed-loop system. Similar arguments as in Siciliano and Khatib (2008) can be utilized to prove that

$$\lim_{t \to \infty} z(t) = 0 \quad (31)$$

provided that $v_d(t) \neq 0$ and $\delta_d(t)$ are uniformly continuous.

The control inputs for the trajectory tracking control are obtained by substituting (22b) and (29) into (21).

Note that the tracking error is only stabilized asymptotically if $v_d \neq 0$, which is not the case at the direction switching points and the parking position. Therefore, we cannot guarantee asymptotic stability. However, the tracking error is monotonically decreased as long as $v_d \neq 0$.

Table 1. Parameters of the car.

l_c	w_c	L	v_m	δ_m	a_m	ζ_m
4.7 m	1.8 m	2.7 m	1 m/s	40°	0.5 m/s^2	20 °/s

Table 2. Geometric configuration of the parking scenarios.

	a	b	d_s	α
parallel	5.5m	2.2m	4m	0°
garage	2.3m	5m	7m	0°
angle	4.7m	2.3m	5m	45°

Table 3. Parameters for the trajectory generation and controller.

r_{χ_2}	2.5
r_κ	10
$k_{1:5}$	$[0.2, 0.53, 0.27, 0.55, 2.75]^T$

Extensive simulation studies confirm that the deviation from the desired trajectory at the parking position for typical parking manoeuvres is considerably small.

Siciliano and Khatib (2008) present controllers for fixed positions, which drive the error to zero even for $v_d = 0$. These controllers apply an additional oscillating velocity to force the car towards the desired position. As the distance to the obstacles is very small for typical parking manoeuvres, these additional velocity inputs may lead to collisions and are therefore not suitable for the problem under consideration.

6. SIMULATION RESULTS

To verify the computational efficiency of the motion planner and the effectiveness of the feedback controller, simulation studies for three different parking scenarios are carried out in the following. The dimensions and dynamic constraints are chosen similar to a mid-sized commercial vehicle. These parameters are shown in Table 1, where l_c represents the length and w_c the width of the car. The maximum steering angle speed is set to $\zeta_m = 15°/\text{s}$, which is below the actual physical limit, in order to provide some reserve for the feedback controller. The geometric configuration of the parking scenarios is summarized in Table 2 with the notation according to Figure 1. The dimensions of the parallel parking lot are chosen such that an additional switching point inside the parking lot is needed. For garage parking another obstacle outside the parking lot is added.

The weighting terms for the optimization problem of the trajectory generation as well as the control parameters are given in Table 3, where $k_{1:5} = [k_1, \ldots, k_5]^T$.

All scenarios are simulated in MATLAB/SIMULINK on an Intel Core i7 machine. The optimization problems are solved with the SQP solver of the numeric software package SNOPT, see (Gill et al., 2006). To solve the OCP (15) for the path-following, a full discretisation is applied beforehand.

The result of the geometric path planner is illustrated in Figure 5. Feasible paths for all scenarios are obtained in less than 15ms computation time.

The path-following problem calculates a solution between two switching points in less than 50ms. The desired control input $\mathbf{u}_{A,d}$ is printed in Figure 6.

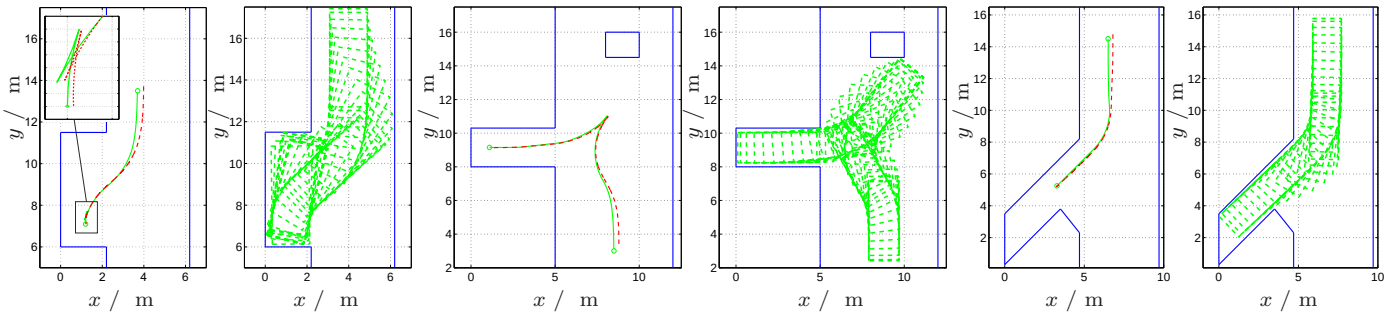

Fig. 5. Planned (solid line) and executed (dashed line) trajectory of the reference point P_R as well as executed trajectory with car boundaries for parallel, garage and angle parking (from the left to the right).

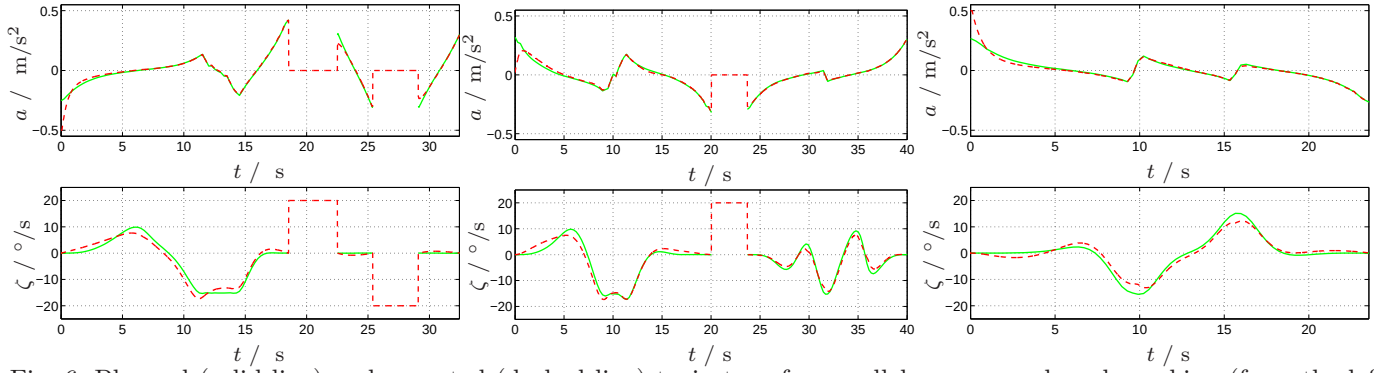

Fig. 6. Planned (solid line) and executed (dashed line) trajectory for parallel, garage and angle parking (from the left to the right).

To verify the performance of the feedback controller, an initial error $\bar{\mathbf{e}}(0) = \bar{\mathbf{q}}(0) - \bar{\mathbf{q}}_d(0) = [30\text{cm}, 30\text{cm}, 0°]^T$ is introduced for all three scenarios. This deviation is corrected within a few driving meters leading to errors at the parking position of $|\bar{\mathbf{e}}(s_{E,n_{SP}+1})| \leq [3\text{cm}, 3\text{cm}, 3°]^T$. At the direction switching points, $\mathbf{u}_{A,d} = [0, \pm\zeta_m]^T$ is applied to reach the desired steering angle $\delta_{d,i+1}$ of the next step in a fast manner. The dashed lines in Figure 5 show the simulated trajectories and the dashed lines in Figure 6 refer to the applied control inputs.

REFERENCES

A. P. Aguiar, D. B. Dačić, J. P. Hespanha, and P. Kokotović. Path-following or reference-tracking? An answer relaxing the limits to performance. In *Proc. 5th IFAC/EURON Symposium on Intelligent Autonomous Vehicles*, Lisboa, Portugal, 5.-7. July 2004.

T. Faulwasser, V. Hagenmeyer, and R. Findeisen. Optimal exact path-following for constrained differentially flat systems. In *Proc. IFAC Word Congress*, pages 9875–9880, Milano, Italy, 28. August-2.September 2011.

M. Fliess, J. Lévine, P. Martin, and P. Rouchon. Flatness and defect of nonlinear systems: Introductory theory and examples. *Int. J. of Cont.*, 61(6):1327–1361, 1995.

P. E. Gill, W. Murray, and M. A. Saunders. *Users Guide for SNOPT Version 7: Software for Large-Scale Nonlinear Programming.* Dept. of Mathematics, University of California, San Diego, CA, February 2006.

J. Hermosillo and S. Sekhavat. Feedback control of a bi-steerable car using flatness application to trajectory tracking. In *Proc. American Cont. Conf.*, volume 4, pages 3567–3572, Denver, CO, 4.-6. June 2003.

M. F. Hsieh and Ü. Özgüner. A parking algorithm for an autonomous vehicle. In *Proc. IEEE Intelligent Vehicles Symposium*, pages 1155–1160, Eindhoven, Netherlands, 4.-6. June 2008.

K. Kant and S. W. Zucker. Toward efficient trajectory planning: The path-velocity decomposition. *The Int. J. of Robtotic Research*, 5(3):72–89, 1986.

M. Khatib, H. Jaouni, R. Chatila, and J.P. Laumond. Dynamic path modification for car-like nonholonomic mobile robots. In *Proc. IEEE Int. Conf. on Robotics and Automation*, number 4, pages 1920–1925, Albuquerque, NM, 25. April 1997.

D. Kim, W. Chung, and S. Park. Practical motion planning for car-parking control in narrow environment. *IET Cont. Theory Applications*, 4(1):129–139, 2010.

M. Krstic, I. Kanellakopoulos, and P. Kokotovic. *Nonlinear and Adaptive Control Design.* John Wiley & Sons, New York, 1995.

J.-P. Laumond, P.E. Jacobs, M. Taix, and R.M. Murray. A motion planner for nonholonomic mobile robots. *IEEE J. Robot. Autom.*, 10(5):577 –593, 1994.

H. Liu, W. Liu, and L.J. Latecki. Convex shape decomposition. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 97–104, San Francisco, CA, 13.-18. June 2010.

T. Lozano-Perez. Spatial planning: A configuration space approach. *IEEE Trans. Comput.*, C-32(2):108–120, 1983.

B. Müller, J. Deutscher, and S. Grodde. Continuous curvature trajectory design and feedforward control for parking a car. *IEEE Trans. Control Syst. Technol.*, 15 (3):541–553, 2007.

C. Nielsen, C. Fulford, and Maggiore M. Path following using transverse feedback linearization: Application to a maglev positioning system. *Automatica*, 46(3):585–590, 2010.

B. Siciliano and O. Khatib. *Springer Handbook of Robotics.* Springer, Berlin, 2008.