Preprints of the 19th World Congress
The International Federation of Automatic Control
Cape Town, South Africa. August 24-29, 2014

Autonomous Parking using a Highly Maneuverable Robotic Vehicle

Alexander Schaub * **Juan Carlos Ramirez de la Cruz** **
Darius Burschka ***

* *Institute of System Dynamics and Control, Robotics Mechatronics Center - DLR, 82234 Wessling, Germany (email: alexander.schaub@dlr.de)*
** *Institute of Robotics and Mechatronics, Robotics Mechatronics Center - DLR, 82234 Wessling, Germany (email: juan.ramirezdelacruz@dlr.de)*
*** *Department of Computer Science, Technische Universität München, 85748 Garching bei München, Germany (email: burschka@cs.tum.edu)*

Abstract: This work presents a novel autonomous parking concept for a four wheel-steerable robotic electric vehicle called ROboMObil (ROMO). Its extraordinary maneuverability, including rotations and lateral driving, and its capability of autonomous driving is demonstrated in the application of parallel parking. The whole process is described starting with the perception to identify a suitable parking spot in the near environment, planning the maneuver, and executing it autonomously. The proposed parking method is free of assumptions and does not use any information about the environment from an external source. Test results with the real vehicle are presented to show the feasibility of the concept.

1. INTRODUCTION

The ROboMObil (ROMO), see Figure 1, is a robotic electric vehicle developed at the Robotics Mechatronics Center of the German Aerospace Center (DLR). This prototype demonstrates how future electric vehicles can benefit from established robotic techniques - see Brembeck et al. [2011]. The ROMO is equipped with four wheel robots, each with its own wheel-hub motor, brake, damper, and steering actuator. The in-wheel steering is not realized by conventional steering mechanics but by an electric drive that was developed for robotic applications. This provides the advantage to steer 35° in one and even 95° into the other direction. The consequence is a high maneuverability including rotations around variable center points and lateral driving. The main human-machine input device is a three degrees of freedom sidestick. The ROboMObil is built according to a four module concept. The first module is the body containing the cockpit, second is the battery mounted beneath the cockpit, and two axle modules which respectively contain two wheel robots and the power electronics.

A central hierarchical control structure ensures the execution of the demanded behavior in all subsystems and provides smooth transitions between the different control modes. The control modes vary in two properties. As described by Schaub et al. [2011] the first choice is between in-vehicle driving or teleoperating. The second choice is the degree of autonomy that can be set from manual control over shared autonomy, as established in space robotics, through to fully autonomous driving.

Autonomous driving is one of the main research fields in the ROboMObil project. A basic requirement for this is the capability of environmental perception. For this purpose, the ROMO is equipped with 18 cameras, which provide a 360° stereo coverage. Cameras have several advantages like their high information density, passivity, low energy consumption, theoretically unlimited range, and ease of integration. The high computational effort is alleviated by a sensor attention management system and the use of parallel computational devices (e.g. Field-programmable gate array - FPGA) for image processing. Nevertheless, the environmental perception is very challenging and will be discussed later in this paper.

The strength of such a robotic vehicle concept can be demonstrated best in a scenario that on the one hand emphasizes its high maneuverability and on the other hand

Fig. 1. The DLR's ROboMObil

is a tedious task if driven manually. Parking space is becoming a rare good in inner cities. Even if the dimension of the space is sufficient for the car, it is not guaranteed that the driver will succeed in parking there. Parallel parking is the perfect scenario to show the different motion modes of the ROMO and its autonomous driving abilities.
In 1989 Volkswagen demonstrated the first automatic parallel parking with their IRVW prototype (see thepetrolstop.com [2013]). This prototype was able to steer all four wheels by several degrees into the same direction. After it was aligned parallel to the parking spot manually, it moved autonomously transversally while alternating between a forward and backward motion. After a few iterations the vehicle was in the parking spot. Since then several approaches for autonomous parking and especially parallel parking for front steered vehicles were proposed. They can be divided into two categories.
The first kind uses computational intelligence methods to learn from a human driver how to get into the parking spot. Daxwanger and Schmidt [1995] proposed two neural control architectures. One consists of a single neural network and the second is a fuzzy hybrid controller, which means the combination of an artificial neural network with a fuzzy network. An approach based solely on fuzzy logic is proposed in Cabrera-Cosetl et al. [2009]. The algorithms of the second category plan the whole motion sequence from a starting point to the goal position in advance. Xu et al. [2000] uses quintic polynomials for the motion planning. Gupta et al. [2010] calculate the trajectory geometrically with the Ackermann steering calculations. Nearly all mentioned approaches use cameras as perception sensor. Only Gupta relies on ultra sound due to the lower costs.
Nowadays, automatic parking is a well known feature in series production vehicles. It was offered first in the 2003 Toyota Prius. Standard parking assistant systems require the driver to pass the parking spot while ultra sound sensors scan the left and the right side of the car. If a search is successful the driver can start the parking maneuver, whereas he has to control throttle and brake while the assistant system controls the steering. Such assistant systems are constantly evolving. In 2013 Volvo demonstrated a solution for driverless automatic parking with obstacle avoidance - see theautoblog.com [2013]. Nevertheless, for parallel parking the parking spot has to be larger then the vehicle's length due to the kinematic constraints of a conventional front-steered car. Additionally, all assistant systems require assumptions like a parallel alignment to the parking spot or external information, e.g. about the location of the goal position. In comparison, the ROMO acts fully autonomously by searching and driving to a parking spot purely based on information derived from its cameras. To the best of the authors' knowledge, this article represents one of the first contributions to the autonomous parking of four-wheel-steered vehicles.
The remainder of this paper is organized as follows. The second chapter describes how the 3D point cloud is segmented to identify objects and how fractured clusters are merged in 2D. After that chapter 3 explains the search for an appropriate parking spot, the maneuver planning, and how the vehicle's hybrid motion dynamics are handled. Chapter 4 shows the results of the ROboMObil parking tests. Finally, conclusions are drawn and an outlook is given in Chapter 5.

2. ENVIRONMENT INTERPRETATION

2.1 3D Segmentation

The segmentation process has the principal objective of breaking down a rigid 3D-reconstruction image into individual 3D entities. Groups of 3D points are clustered that possibly belong together in the real world. The benefits of this process are twofold. First, we obtain a gross impression on how the scene is spatially structured, which is also required for 2D/3D map building. Second, every single clustered entity, or 3D blob, might have a semantic meaning by its own. This means that the segmented 3D-blobs represent the potential actor candidates in the scene: pedestrians, (part of) cars, buildings, traffic lights, etc. The segmentation procedure occurs as follows. Once a rigid stereo reconstruction is captured, a plane terrain is assumed and it is removed from the 3D reconstruction. This leaves a set of raw 3D points floating on the foreground without any structural relation among any of them. Following Ramirez and Burschka [2011], these spatial connections are found by means of a Depth-First Search (DFS) on the points. After that, each detected 3D blob is encapsulated in a box whose dimensions are determined by the Principal Component Analysis (PCA) method. As a final result, we obtain a set of 3D blobs representing tentative actors. For each of them the dimensions, pose consisting of position and orientation, and semantic label are determined (see Figure 4(d)).

2.2 Blob Merging

Even a dense 3D reconstruction algorithm, such as the semi-global matching used here (see Hirschmüller [2008]), is not able to provide a depth value for every pixel in an image. The reason can be e.g. homogeneous textures or reflections. Vehicles tend to have both properties and therefore a parking car is not always reconstructed as a closed point cloud. The described segmentation algorithm may identify the parking car as several neighboring objects. This is obstructive for the planning algorithm that detects parking spots for the ROMO. Hence, an object merging algorithm was developed to cope with fractured objects. Different to the segmentation the merging works in the 2D space. For the parking it is sufficient to consider only the projection of the objects to the ground plane. The set of raw blobs B_r is received from the segmentation, whereas each blob β^3 is defined by its center of gravity c^3 in 3D coordinates, its expansion e_n in the x,y and z direction, and angle α, which is the rotation of the blob to the world coordinate system. First small blobs, which can be created due to noisy data from the 3D reconstruction, are filtered out if at least one of their expansions is below a certain limit. From now on only the projection to the xy-ground plane is considered $\beta^2 \equiv \beta$. The filtered blob set is ordered according to the size of the expansion. This is done to simplify the merging process. The algorithm starts with the widest blob and goes through the descending list, since larger blobs will attract smaller ones. If a blob is already attached to another for merging, it will be omitted when searching for aggregation candidates for other blobs.
In order to see if two blobs, $\beta_1 \& \beta_2$, have to be merged, the smaller one β_2 is aligned with the orientation of the wider

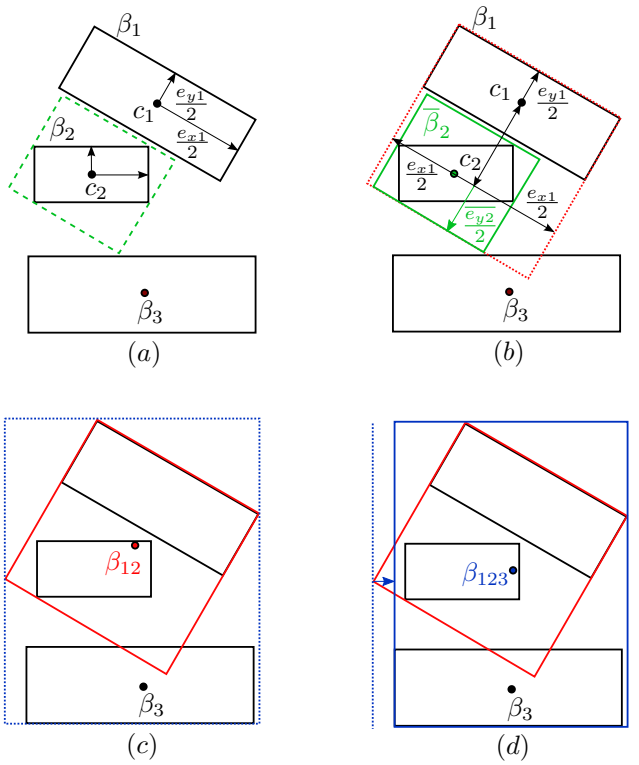


Fig. 2. (a) Rotating β_2, (b) Merging $\beta_2 \cup \beta_1$, (c) Merge Error of $\beta_{12} \cup \beta_3$, (d) Optimization of β_{123}

blob β_1. See β_2 in Figure 2a is transformed to $\overline{\beta_2}$ in Figure 2b. After that the following criterion has to be fulfilled:

$$\sqrt{(c_{x1}-c_{x2})^2+(c_{y1}-c_{y2})^2} - \mathrm{d}_{lim} \leq \sqrt{(\frac{e_{x1}}{2})^2+(\frac{e_{y1}}{2})^2}+\sqrt{(\frac{e_{x2}}{2})^2+(\frac{e_{y2}}{2})^2} \quad (1)$$

The clearance parameter d_{lim} could be set to increase the merging probability, but a certain safety distance is already contained in the criterion (1) - see Figure 2. An exception is the case when two corners are the nearest points. This could be neglected since the reconstruction is very accurate at corners and edges and the segmentation has its own safety clearances. The criterion is also fulfilled if one blob is covered by the other in at least one direction - see $e_{x1} = e_{x12}$ in Figure 2b. This must be checked first as the expansion in e_{n12} of the merged blob β_{12} would be equal the larger of e_{n1} or e_{n2}. If this is not the case, the new expansion in direction n is:

$$e_{n12} = \frac{e_{n1}+e_{n2}}{2} + \|c_{n1}-c_{n2}\| \quad (2)$$

Now it is possible to calculate the new center of gravity. Therefore, the root blob that has the largest x value in the coordinate system and the root blob that has the highest y value are determined.

$$c_{n12} = c_{nw} + \frac{e_{nw}}{2} - \frac{e_{n12}}{2} \quad (3)$$

The index w stands for the number of the blob with the largest value in the respective direction. Since the center of gravity must be expressed in the world coordinate system c_{12} has to be rotated by α_{12}, which is equal the angle of the dominant root blob. The root blobs are now removed from the current set of blobs B_c. The list of initial blobs is static because this information is needed for the optimization step. It can occur that blobs, which were merged more than once, i.d. have more than 2 root blobs, expand too far (see Figure 2c). After the merging step a blob is optimized by comparing its expansions to the area covered by root blobs from B_r. If a merging error has occurred the expansion is corrected and the center of gravity is shifted. Compare Figure 2c to Figure 2d. After this optimization step the merged blob is added to B_c. With real world data the optimization occurs very seldom, as on the one hand more than two merging steps are necessary and on the other hand the angle of the merged blob must change over the different steps. The merging stops if the algorithm iterates over B_c without any further merge. The final set of blobs is then passed to the planning algorithm.
It should be remarked that the merged blob of the artificial example of Figure 2 is a coarse approximation of the size of the real objects. A more accurate merge would be possible if the structure of the merged blob was not bound to the rectangular description of the root blobs. Nevertheless, this representation is sufficient for the parking application.

3. PLANNING THE MANEUVER

The processed 2D blobs are passed to the planning algorithm in order to find a parking spot and plan the trajectory for the maneuver without any assumptions or prior knowledge about the environment. For this planning algorithm the origin of the coordinate system lies in the center of the vehicle R that should park autonomously. The orientation of the coordinate system's x-coordinate is aligned with R's driving direction and the y-coordinate points to the left side - see Figure 3.
First the algorithm goes through the object list Ω to find neighboring pairs with a valid parking spot between them. A pair of objects is saved as neighbors if no other object lies between them. For determining the neighbors of O_n all vectors that start from c_n to all other centers of gravity are calculated. If the direction of one vector is similar to another, the one with the shorter distance is the direct neighbor. This simplification is possible due to the merging step. To better illustrate this claim, let us consider an object that is very thin in one direction and very large in the other. If it lies like a wedge in between two other objects than the criterion with the center of gravities might not work, but the 'wedge' would have been already merged with at least one of those blobs.
Each object $O_n \in \Omega$ is generated from the respective blobs. Additionally to the blob description the four corners k^n_{1-4} of the object O_n are calculated and added, as they are needed to identify facing edges of the objects. In Figure 3 the distances between all corners of O_1 and all corners of O_2 are calculated. The two shortest connections are determined under the condition that each corner is selected not more than once. These are the segments $\overline{k^1_2 k^2_4}$ and $\overline{k^1_1 k^2_3}$. They denote the boundaries of the parking spot and therefore have to be larger than the vehicle plus a safety margin. If this is true the supporting points k^1_5 k^2_5 are generated as the middle points of the facing edges. Objects surrounding a parking spot might not be perfectly aligned in reality, but the direction of the vector connecting k^1_5 and k^2_5 will provide an acceptable target orientation α in any case. This angle can be calculated as:

$$\alpha = \arctan 2(\overline{k^1_5 P_2}^{\,y}, \overline{k^1_5 P_2}^{\,x}) \quad (4)$$

The center of the parking vehicle R should reach the final position denoted by P_2, which is the middle point of $\overline{k_5^1 k_5^2}$. The goal position G is the combination of P_2 together with the target orientation α. At the beginning of the maneuver R is located at P_0. While it is not always possible to drive directly to G, a highly maneuverable vehicle like the ROboMObil can achieve this in at most three steps.
The first mode will be a longitudinal motion from P_0 to the intermediate point P_1. This point should be calculated as such that the ROMO can rotate to the target orientation and then drives laterally to P_2. Therefore, P_1 lies on a line through P_2 and perpendicular to $\overline{k_5^1 k_5^2}$. Moreover, the side, on which P_1 is located, has to be selected as well, as P_2 can be approached by crossing either $\overline{k_1^1 k_3^2}$ or $\overline{k_2^1 k_4^2}$ in Figure 3. Naturally, the closer segment is chosen if it is not blocked. The distance to from P_1 to P_2 is chosen so that a clearance of at least half of the ROMO's length to the corners k_1^1 and k_3^2 is given. This is necessary for a safe rotation of the ROMO at P_1. A perpendicular approach from P_1 to P_2 is not always possible even if the parking lot is large enough. This could occur if the objects spanning the parking lot are oriented extremely inclined. The algorithm detects this and shifts P_1 so that $\overline{P_1 P_2}$ is perpendicular to the parking spot's boundary that will be crossed - e.g. $\overline{k_1^1 k_3^2}$ in Figure 3. When this checking and processing has been applied to all possible combinations of objects, one parking spot is selected. Finally, the set of selected target points (P_1, P_2) is passed to the trajectory calculation algorithm.

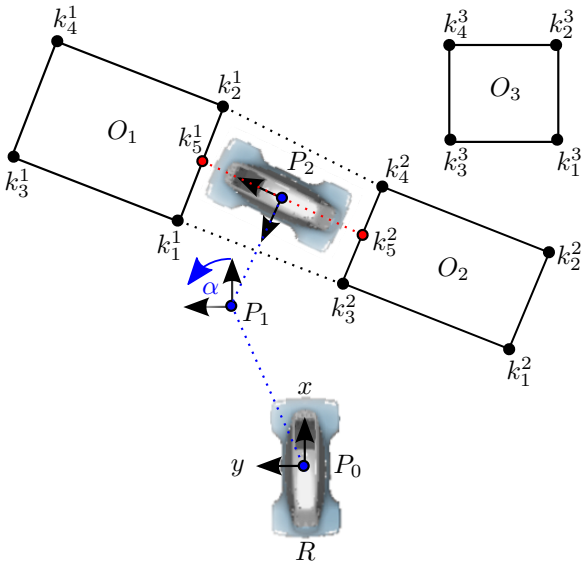


Fig. 3. Planning the Parking Points

The type of the trajectory is dependent on the motion mode. The ROMO will move in all of its three different modes to reach the goal position. A detailed description of the different motion control modes and geometric constraints can be found in Bunte et al. [2011]. We focus on the description of the interface from the autonomous planning module to the vehicle dynamics control here. This interface allows the motion of the ROMO's geometrical center and its orientation to be commanded by the trajectory planner, rather than having to directly command individual wheel speeds and steering angles on each of the four wheels.
A trajectory for the longitudinal mode is described by:

(1) the velocity v, acceleration a, or $\dot{a}$ in the tangential direction
(2) path curvature ρ
(3) side slip angle β.

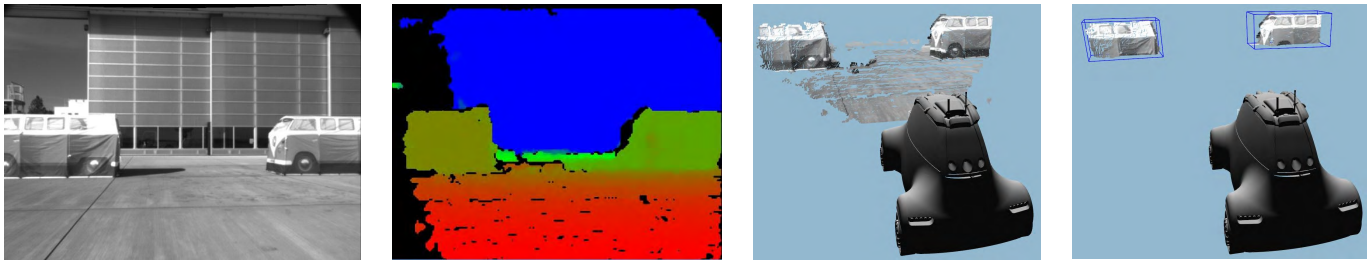
The speed of the ROMO can be described using either v, a, $\dot{a}$ or a combination of them. This depends on the chosen controller in the vehicle dynamics module. A special feature of the ROMO is that the curvature ρ and side slip angle β can be commanded separately. In a conventional vehicle with front wheel only steering β is not directly controllable. Additionally, the trajectory can also include the derivatives $\dot{\rho}, \dot{\beta}$ if the selected controller supports those inputs e.g. for feedforward control.
A totally different set of parameters is chosen to describe the motion in the rotational mode:

(1) x-coordinate of the instantaneous center of rotation (ICR) x_{ICR}
(2) the ICR's y-coordinate y_{ICR}
(3) the yaw-rate $\dot{\psi}$

The x_{ICR} and y_{ICR} are not fixed to ROMO's center as also rotations around external points are possible. For a detailed explanation of the ICR's geometrical limitations see Bunte et al. [2011]. Again it is also possible to pass the derivatives $\dot{x}_{ICR}, \dot{y}_{ICR}, \ddot{\psi}$ to vehicle dynamics controller.
The lateral motion is very similar to the longitudinal with the main difference that $\beta = 90°$ is fixed. Even during the lateral motion the curvature can be varied due to the maximum steering angles of 95°.
The parking maneuver can be planned in a straightforward manner in accordance with this interface. The best motion control mode for the first phase, in which the ROMO has to drive to P_1, is longitudinal driving. It is important that the ROMO's center will stop at P_1. The orientation in this phase is of minor importance, but it is desirable that the rotation is kept to a minimum for the perception of the parking spot. Consequently, the algorithm initially attempts to build a trajectory only changing β while the curvature is fixed $\rho = 0$. Before planning the feasibility of such a trajectory must be tested with this criterion:

$$\beta_{max} \leq \arctan 2(P_1^y, P_1^x) \tag{5}$$

If the side slip angle is over its max value the trajectory has to be calculated differently. A good choice is the construction of a clothoid to P_1 using the method proposed in Wilde [2009]. The speed profile for this phase is computed so that the ROMO stops at P_1, where a change in motion control mode is executed. It is only possible to switch between motion control modes when the ROMO is in standstill. The calculation is solved analytically with taking maximum values and rate limitations into account.
The next phase is the rotation to align the ROMO in parallel to the parking spot. Since the center of the vehicle is at P_1, we can keep $x_{ICR} = 0$ and $y_{ICR} = 0$. The ROMO has to cover an angle $\psi_{dem} = \alpha_{cur} - \alpha$. The trajectory of $\dot{\psi}$ is calculated as a simple integration. The constraints here are also stillstand at the end and maximum velocity and rate values.
Finally, the lateral driving is planned. Due to the choice of P_1 only the velocity trajectory has to be computed. It is an integration of the lateral velocity to cover the distance $\overline{P1P2}$ in the correct direction. Again the conditions apply that the velocity is zero at the end, while velocity and rate



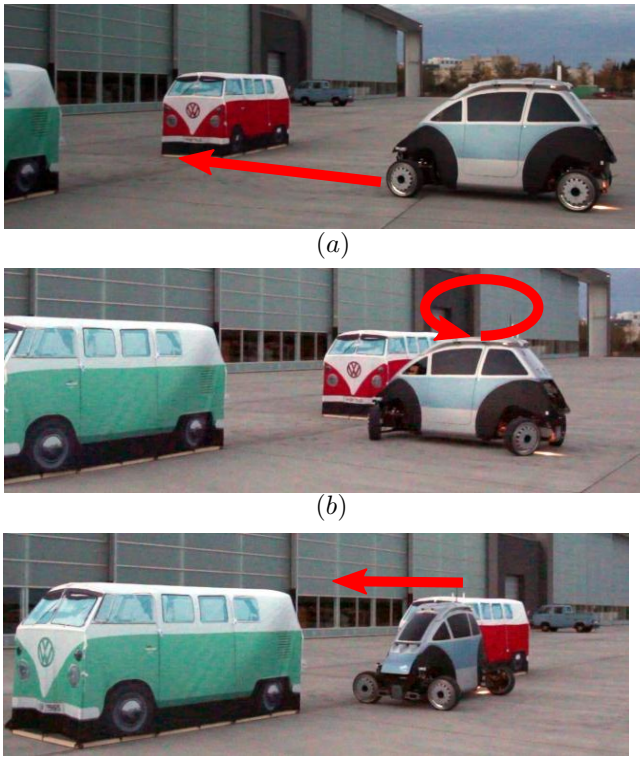
(a) (b) (c) (d)

Fig. 4. (a) Front Camera Image, (b) Disparity Map, (c) Visualization of the 3D Points, (d) Segmented Blobs

limitations have to be obeyed. After completing the lateral driving trajectory the motion mode demand is changed again to longitudinal driving to align the wheels.
If any of these phases are not necessary, the respective modes are not planned. Finally, the created sequence is passed to a trajectory execution module that controls and supervises the autonomous parking maneuver. It is directly connected to the vehicle dynamics control and provides the current motion demand.

4. EXPERIMENTAL RESULTS

This section describes a real parking maneuver in detail. The parking spot is framed with two tents shaped like Volkswagen transporters of the first generation. The ROboMObil's initial position P_0 is approximately 12m in x-direction and 2m in y-direction to the middle of the gap between the two tents. The orientation of the two transporters is about 95°. Figure 4a shows the scene taken from the ROMO's front cameras. For this test we only use this stereo pair as it covers the two objects and the target position sufficiently.
First the perception analyzes the camera images to find a suitable parking spot. A disparity map is generated from the stereo images by the semi-global matching (SGM) (Figure 4b). The 3D point cloud is directly calculated using the disparity values and the segmentation extracts the objects of interest - see Figure 4c & d. Since the disparity map is dense and accurate the visible parts of the tents are not fractured and the merging algorithm is not changing the blobs. Due to our FPGA implementation of SGM the image processing runs with $\approx$ 15 frames per second.
After that the planning algorithm receives the 2D projection of the blobs and calculates the intermediate position $P_1 = (9.2, 1.5)$[m], the rotation angle $\alpha = 1.658$[rad], and the goal position $P_2 = (12.2, 1.8)$[m]. With only two blobs the search of neighboring pairs is trivial and only one valid parking spot is detected.
Now the two points P_1, P_2 are passed to the planning module, where first the trajectory for the longitudinal motion is planned as described in the preceding chapter. The execution of this trajectory begins and the ROMO starts to adjust the side slip angle and moves (Figure 5a). The covered path is determined from odometry or the dGPS aided IMU and fed back to the execution module to determine where on the planned path the ROMO currently is. This feedback is necessary to reach the target positions with a sufficient accuracy. When the ROMO stops at P_1 it triggers again the segmentation of the 3D point cloud to get an update of the blobs and consequently of the



(a) (b) (c)

Fig. 5. Autonomous Parking Maneuver: (a) Longitudinal Mode, (b) Rotational Mode, (c) Longitudinal Mode

parking spot. After the target position is updated the ROMO changes into rotation mode. With the finished initialization of this mode the execution of the rotation trajectory starts - see Figure 5b. Here the feedback of the currently covered angle is necessary to align the properly to the parking lot. Now the ROMO stands at P_1 and is aligned in parallel to the parking spot. The third part of the maneuver begins and the motion mode changes to lateral driving. After it is properly initialized, the ROMO starts to drive laterally into the parking spot (Figure 5c). Due to the rotation the distance to P_2 is now $\approx$ 3 m in negative y-direction . Finally, the motion control mode is changed into longitudinal to align the wheels. During the tests the maximal velocity was $|v_{longMax}| = 4m/s$ in longitudinal and $|v_{latMax}| = 2m/s$ in lateral mode. The accelerations were limited to $|a_{longMax}| = 0.5m/s^2$ and $|a_{latMax}| = 0.5m/s^2$. For the rotational mode the limits were $|\dot{\psi}| = 1.2rad/s$ and $|\ddot{\psi}| = 0.5rad/s^2$. The trajectory generation and the vehicle controls run on a

realtime system with 4ms cycle time.
Tests from different positions and setups showed that an open loop executions of the trajectories results in deviations of 10-20% from the goal position. This is not acceptable for the parking application. The local coordinate system is bound to the ROMO. Hence, a feedback of the covered path is an easy and effective solution for position control.
Another interesting point is the tuning of the vehicle dynamics controllers. At the beginning of the parking tests the ROMO was using parameter sets for the speed controllers that were designed for manually driving. The results were deviations of 10-15% from the target position in longitudinal and lateral mode, since the controller parameters were set up with a small proportional value and no integral value. Notably better results (<1% deviation at the goal position) were obtained with a fundamental different parameter set especially tuned for autonomous driving. For the longitudinal and lateral mode the proportional value was set 5 times higher and the integral value was activated, so that the controller was fast enough to follow the precise trajectory generated by the artificial intelligence. A human driver is not able to command such precise motion demands and hence using this setup for manual driving mode would lead to a 'shaky' vehicle behaviour.
At the current state the perception is executed twice. Although the blobs are observed and averaged over several time steps the whole maneuver is very sensitive to incorrectly detected blobs at the beginning. A continuously running measurement and replanning could increase the robustness of the perception.

5. CONCLUSIONS AND FUTURE WORK

This paper introduced the ROboMObil's autonomous driving capability and demonstrated it on the example of parallel parking. The proposed scheme is free of assumptions regarding the initial pose and goal position and does not require any information about the location of the parking spot and the environment from an external source. When triggered it searches the environment for a suitable parking spot and parks there autonomously in at most three steps. The high maneuverability reduces the minimal required size of the parking spot, which would be very welcome in inner cities. But also applications in the logistics sector or for search and rescue missions are conceivable for an autonomous vehicle with the size and maneuverability of the ROboMObil.
The artificial intelligence capabilities of the ROMO will continue to be extended. Several possible extensions based on the parking maneuver are planned, and the parking task could be part of a larger navigation maneuver, where the ROMO drives around for a while and builds a local map based on the segmented 3D data. Furthermore, the planning scheme could be extended to freely select the sequence of motion modes. Currently, the ROMO has to follow the described sequence with the possibility to omit steps. Moreover, a technique from vision based control could be applied to align the ROMO more accurately in the parking spot.

ACKNOWLEDGEMENTS

The authors would like to thank the German Aerospace Center (DLR) for funding this project, Johann Bals, Jonathan Brembeck, and the ROboMObil team, who built the test platform and supported the experimental validation.

REFERENCES

Jonathan Brembeck, Lok Man Ho, Alexander Schaub, Clemens Satzger, and Gerhard Hirzinger. Romo - the robotic electric vehicle. In *22nd International Symposium on Dynamics of Vehicle on Roads and Tracks*. IAVSD, 2011.

Tilman Bunte, Jonathan Brembeck, and Lok Man Ho. Human machine interface concept for interactive motion control of a highly maneuverable robotic vehicle. In *Intelligent Vehicles Symposium (IV), 2011 IEEE*, pages 1170 –1175, june 2011. doi: 10.1109/IVS.2011.5940490.

R. Cabrera-Cosetl, M.Z. Mora-Alvarez, and R. Alejos-Palomares. Self-parking system based in a fuzzy logic approach. In *Electrical, Communications, and Computers, 2009. CONIELECOMP 2009. International Conference on*, pages 119–124, 2009.

W.A. Daxwanger and G.K. Schmidt. Skill-based visual parking control using neural and fuzzy networks. In *Systems, Man and Cybernetics, 1995. Intelligent Systems for the 21st Century., IEEE International Conference on*, volume 2, pages 1659–1664 vol.2, 1995.

A. Gupta, R. Divekar, and M. Agrawal. Autonomous parallel parking system for ackerman steering four wheelers. In *Computational Intelligence and Computing Research (ICCIC), 2010 IEEE International Conference on*, pages 1–6, 2010. doi: 10.1109/ICCIC.2010.5705869.

H. Hirschmüller. Stereo processing by semiglobal matching and mutual information. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(2):328 –341, 2008. ISSN 0162-8828. doi: 10.1109/TPAMI.2007.1166.

J.C. Ramirez and D. Burschka. Framework for consistent maintenance of geometric data and abstract task-knowledge from range observations. In *Robotics and Biomimetics (ROBIO), 2011 IEEE International Conference on*, pages 963–969, 2011. doi: 10.1109/ROBIO.2011.6181412.

Alexander Schaub, Jonathan Brembeck, Darius Burschka, and Gerd Hirzinger. Robotic electric vehicle with camera-based autonomy approach. *ATZelektronik*, 2(2): 10–16, April 2011.

theautoblog.com. 2013. URL `http://www.autoblog.com/2013/06/20/volvo-demos-autonomous-self-parking-car-concept`.

thepetrolstop.com. 2013. URL `http://www.thepetrolstop.com/2013/05/vw-irvw-futura.html`.

D.K. Wilde. Computing clothoid segments for trajectory generation. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 2440 –2445, oct. 2009. doi: 10.1109/IROS.2009.5354700.

Jin Xu, Guang Chen, and Ming Xie. Vision-guided automatic parking for smart car. In *Intelligent Vehicles Symposium, 2000. IV 2000. Proceedings of the IEEE*, pages 725–730, 2000. doi: 10.1109/IVS.2000.898435.