Lidar-based Mapping and Localization for Autonomous Racing

Markus Schratter,[1] Jasmina Zubača,[2] Konstantin Mautner-Lassnig,[3] Tobias Renzler,[2] Martin Kirchengast,[2] Stefan Loigge,[1] Michael Stolz,[1,2] Daniel Watzenig[1,2]

***Abstract*— Autonomous racing challenges, since the very beginning of automated driving, have inspired new development and defined state of the art. By this, autonomous racing has strongly contributed to the research field of automated driving and indirectly generated important societal impact, increasing traffic safety by reducing or even avoiding human errors in driving. Accurate, reliable and robust perception is a key factor for driverless vehicles, and lidar sensor is currently one of the most promising sensor technologies used for environmental perception. A lidar sensor is versatile as it can be used for mapping purposes, object detection and for the localization. This makes it predestined for many automotive applications. In this publication, the *Autonomous Racing Graz* team reveals their approaches for mapping and localization using only a lidar sensor, and the modified software components from Autoware. Developed solutions are applied and validated in two ROBORACE Season Alpha challenges. The entire process, from mapping, over path planning, to online localization, is summarized and discussed.**

I. Introduction and Related Work

Autonomous driving requires a permanent monitoring of the environment, that includes detection, recognition, and classification of objects [1]. Based on these, the vehicle needs to respond accordingly, in a tactical or operational way, e.g., lateral and longitudinal motion control. Therefore, accurate environmental perception, robust vehicle state estimation and precise localization of the vehicle within a map are highly important. Also, to increase the response quality even further, the prior knowledge of the road geometry can be used as well. This is essential for racing, as it allows to reach the limits of high speed driving [2].

Maps that allow precise localization can be generated using GPS or lidar sensors [3]. While GPS relies on infrastructure and is not always available, lidar allows a standalone localization based on reflectivity or Gaussian mixture maps [4], [5]. Also, simultaneous localization and mapping (SLAM) is possible: building a consistent map of the environment while simultaneously determining the location within the map [6].

ROBORACE (RR) is a competition of autonomously driving, electrically powered vehicles: it requires from the participating teams to master given tasks to the best possible extent [7]. In the so called *precision* and *localization* challenges, a circuit has to be driven as fast and accurate as possible. The racetracks and the environment has a high variation in the layout and the structure. The track contains tight gates, as shown in Fig. 1, that have to be passed at high speeds. To the net driving time, penalties, e.g., missing a gate or hitting cones, are added. The prior usage of lidar technology within RR is given in [8], where combined IMU, GPS and lidar measurements are fused for localization, to meet the high speed requirements. In comparison to RR racetrack setup, Formula Student Autonomous Racing Series uses well marked racetracks with the cones, and an overview for mapping and racing can be found in [9]. Also, reports about lidar-based localization and dead reckoning are given in [10], while a general survey of sensor technologies and the state of the art is described in [11].

Fig. 1. The Roborace DevBot passes a gate on the Circuit de Croix-en-Ternois in France. The lidar is placed on the top of the race car.

Although there is a lot of research in previously mentioned fields, there is no prior work analyzing and evaluating the entire process, starting from a high precision mapping to online localization for racing, where only a limited time for the whole process on a racetrack is available. The proposed approach adapts and extends known methods to meet the challenging requirements of high-speed driving and enables accurate, robust, and low feature lidar localization. Further, its performance is compared with GPS based localization.

The article is structured as follows: After introducing the challenges of the competition in Section II, the mapping process with the racing line generation is explained in Section III. The localization algorithm, its required software stack and used time synchronization mechanism are described in Section IV. The proposed approach is evaluated in two competitions in Section V.

II. Problem Definition

The gates placed during the challenges within RR's Season Alpha have a width of 2.3 m. With a vehicle width of 2.0 m, only 0.15 m on each side is available to avoid collisions with the cones that represent the limits of the gate. The three main challenges arise: First, lidar-based

[1]Markus Schratter, Michael Stolz, Stefan Loigge and Prof. Daniel Watzenig are with the Virtual Vehicle Research GmbH, Graz 8010, Austria, markus.schratter@v2c2.at.

[2] Jasmina Zubača, Tobias Renzler, Michael Stolz, Martin Kirchengast and Prof. Daniel Watzenig are with Institute of Automation and Control of Graz University of Technology, Graz 8010, Austria, {jzubaca,tobias.renzler}@tugraz.at.

[3] Konstantin Mautner-Lassnig ist with ARTI - Autonomous Robot Technology GmbH, Goesendrf 8077, Austria, k.ml@arti-robots.com.

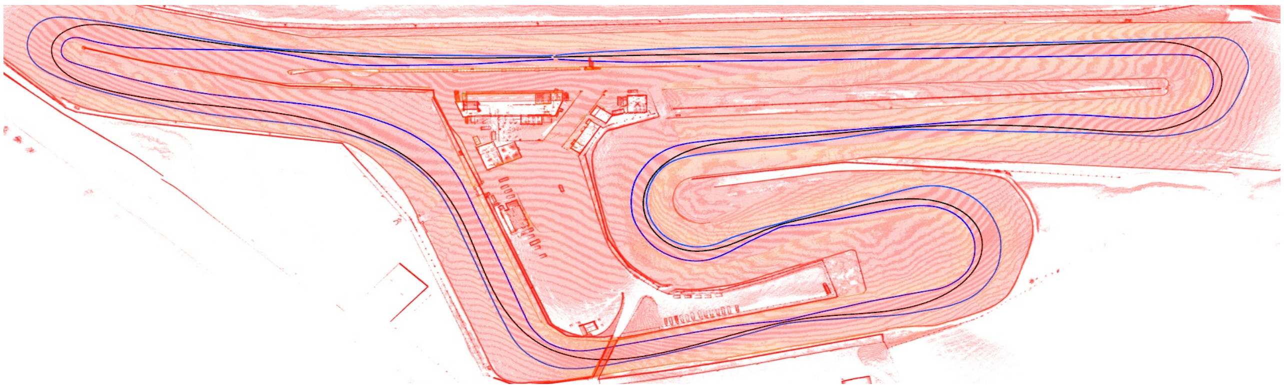

Fig. 2. Circuit de Croix-en-Ternois in France. Point cloud data are red. Track borders are blue and the racing line is black.

localization, GPS independent, needs to be robust, provide high update rates, and account for high accelerations. Second, computation time, sensor inaccuracies, and synchronization between components that may not be worth of mentioning under normal driving, become highly critical under racing conditions. Third, due to the nature of a racing circuit, the map contains a reduced amount of features, compared to inner-city driving: information about restricted areas, gate positions, and path planning must be obtained from the generated lidar map.

III. Mapping

The progress of making a map of the operating area as a racetrack is even today a tedious and not always well-defined procedure: especially, when environmental factors like fences or hard to detect materials come into play. One has to find the ideal setup between map size and quality due to the amount of data which further affects the processing speed. Nevertheless, a full 3D map of the area can provide major advantages in several disciplines. The localization uses the full extent of the available map information and outputs more precise and stable position. When having a look at the rich point cloud data in every dimension, it appears obvious that a 3D approach is superior among any other 2D method. The 3D point cloud has also the advantage that the ground of the mapped area is represented and the reflectivity of materials (e.g. asphalt, vegetation, cone, etc.) are included. The knowledge about exact borders of the racetrack increases the robustness and quality in raceline computation. The mapping is done in an offline manner by recording the area of interest and running the Normal Distributions Transform (NDT) mapper afterwards. The offline computation enabled testing of different parameter configurations to obtain the best mapping result. The technique behind NDT is described in Section IV.

A. Point cloud map

The applied mapping algorithm outputs a 3D point cloud, where every point consists of position and measured intensity. The intensity represents the reflectivity of a measurement and depends on the traveled distance and the underlying material. Using the intensity for visualization, the differentiation between road, vegetation and other objects can be further increased. Fig. 2 visualizes the point cloud map consisting of 4.5 million points of the racetrack in France.

B. Transformation between coordinate systems

Lidar localization is always calculated with respect to the previously recorded high definition map (local coordinates within the map). However, if the racing trajectory is defined in global coordinates, tracking needs the actual position in global coordinates during the race as well. To derive the required static transformation from lidar (local) position to global position upfront, two different approaches can be taken:

(A) Calculating the transformation from matching distinct reference points (landmarks) known in advance (e.g., via Google maps) during mapping via CloudCompare, an open source cloud and mesh processing software [12]. The advantage is that no GPS measurement is needed during the map recording. The disadvantage of this approach is the required knowledge of exact positions of landmarks.

(B) In contrast to this, the optimization based approach used by the *Autonomous Racing Graz* team is summarized next. The idea is to use already available sensor measurement traces including GPS measurement from test drives. Compared to approach (A) this method relies on GPS measurement (disadvantage), but does not need prior knowledge of distinct landmark locations upfront (advantage). The transformation parameters (rotation angle α and translation $[d_x, d_y]$) are found by optimizing the matching between transformed lidar measurement and corresponding GPS measurement. This is done in two steps: (a) a least squares (LS) approach to find an initial transformation and (b) an iterative closest point (ICP) approach to fine-tune [13]. Step (a) is beneficial, since step (b) should be initialized in close vicinity of the final solution. Note that step (a) minimizes the Euclidean distances between related measurement samples, while step (b) matches the shapes of the entire measurement traces in total. The LS is done as follows: The geometric relations between lidar and GPS measurement

$$\begin{bmatrix} x \\ y \end{bmatrix}_{\text{GPS}} = \begin{bmatrix} d_x \\ d_y \end{bmatrix} + \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}_{\text{lidar}} \quad (1)$$

for each time instance are combined to form an over determined linear system $A \cdot X = B$ by involving all time instances as

$$B = \begin{bmatrix} x \\ \vdots \\ y \\ \vdots \end{bmatrix}_{\text{GPS}}, A = \begin{bmatrix} 1 & 0 & x & -y \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 1 & y & x \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{\text{lidar}}, X = \begin{bmatrix} d_x \\ d_y \\ \cos\alpha \\ \sin\alpha \end{bmatrix}. \quad (2)$$

X is obtained by using the More-Penrose pseudo-inverse $X = \left(A^T A\right)^{-1} A^T \cdot B$, which leads to the LS solution for X. From X, the angle α can be calculated by $\alpha = \text{atan2}\,(X_4, X_3)$. Finally, Fig. 3 shows that the ICP-step is able to generate more accurate matching with respect to the LS approach with different cost criteria, i.e., the minimum distance between each GPS measurement sample and all transformed lidar measurement samples.

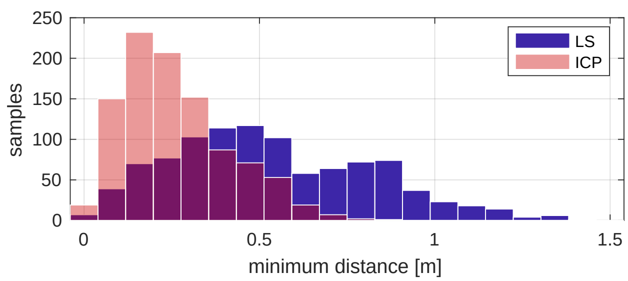

Fig. 3. Histogram of the minimum distances between each GPS measurement and the entire transformed lidar measurement trace for the Zala Zone racetrack. Average minimum distance error for LS: $0.53[m]$, ICP: $0.26[m]$.

C. Racetrack layout

Raceline optimization requires recorded point cloud data enhanced with metadata of the racetrack, i.e., defined borders. The following steps need to be performed. First, lidar measurements are recorded while driving slowly along the track. Second, the recorded lidar measurements are used to generate the point cloud map. Third, the created point cloud map is processed in RVIZ (Visualization tool for ROS) to obtain the point cloud. The metadata of the inner and outer border are defined visually by hand, since the racetracks' boundaries differ from race to race, (cones, boxes, curbs, walls or their combination) and automation does not pay off. As example, the racetrack in Zala Zone (Hungary), with inner and outer borders highlighted in black, is depicted in Fig. 4.

D. Reference line generation

The layout of the racetrack is defined by inner and outer borders, represented by a list of global 2D coordinates. Based on these, a reference line used for raceline optimization is generated. The smoothness of the reference line is important, as it ensures convergence in further optimization.

The reference line generation uses an algorithm developed for fast tuning and adaptation during the racing events. It incorporates narrow gates and also provides low computational effort. The reference line generation consists of eight steps: (1) data reading and duplicates removal, (2) linear interpolation, (3) distance matrix calculation, (4) pair finding, (5) reference line calculation and (6) smoothing, (7) inclusion of the corner

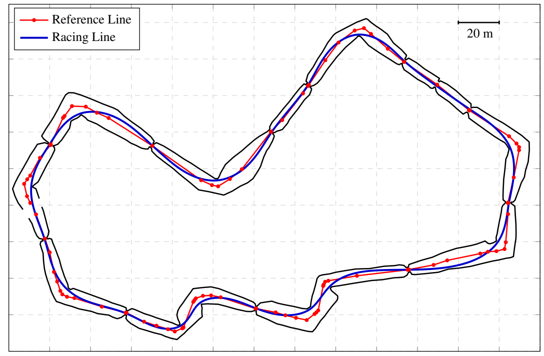

Fig. 4. Racetrack on the proving ground Zala Zone in Hungary. Track borders are black. The reference line is red. The optimal racing line is blue.

points and (8) gate points. The resulting reference line is shown in Fig. 4, highlighted in red. A detailed description of the algorithm can be found in [14].

E. Optimal racing line generation

The smooth reference line is an input to the algorithm that calculates the optimal racing line. The minimum curvature trajectory planning approach, given in [2], is used for raceline optimization. In comparison to [15], this approach achieved better performance in terms of smoother and smaller peak curvature, making it more robust to real world applications. Also, the significant lap time reduction could be achieved. The resulting optimal racing line is depicted in blue, in Fig. 4.

IV. Localization Architecture

A. Normal distributions transform

The Normal Distributions Transform (NDT) algorithm is used for mapping and localization [16]. NDT is a part of the open-source self-driving stack Autoware. In this configuration, a 64 layered lidar was used to record the data for mapping whereas the one with 16 layers for localization. In principle, a real measurement is compared with previously generated point cloud map. The created point cloud map is taken as a reference and further broken down to three-dimensional boxes, with each having individual probability distribution. During the alignment process, the algorithm searches for similarities. Outliers or changes in the environment alter the matching quality and represent a common problem. The applied probability density functions can solve this by enabling a ”near match” to the alignment. Fig. 6 shows both point clouds, that are required for receiving a position, are visualized for comparison.

B. Software components

Autoware is one of many open-source stacks used to address the problems of self-driving vehicles. It runs on the top of the middle-ware ROS and it unites individual software components, providing interfaces and tools for fast and easy setup routines for automated driving [17]. An overview of the software architecture for the mapping and localization is visualized in Fig. 5. The top part shows the required components for the mapping and the bottom part the components for the localization during driving. In both cases, the **os1_node** provides the point cloud from the lidar.

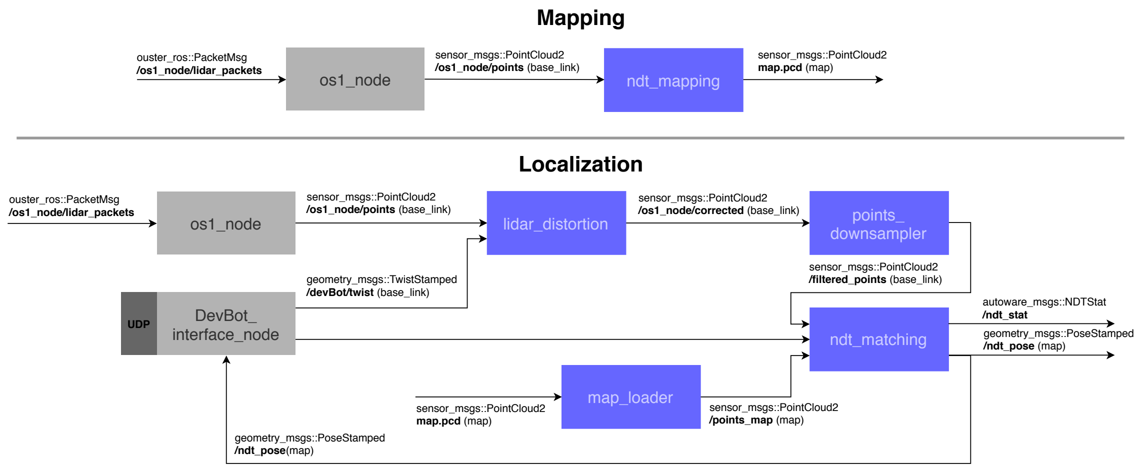

Fig. 5. Software components running on the DevBot 2.0. Top: track mapping. Bottom: lidar localization running online on the autonomous race vehicle.

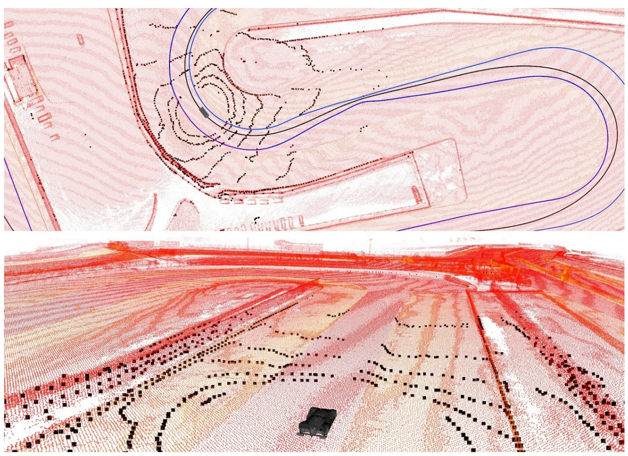

Fig. 6. The point cloud map (red) is used to match real measurements (black) that allow to obtain the location on the racetrack.

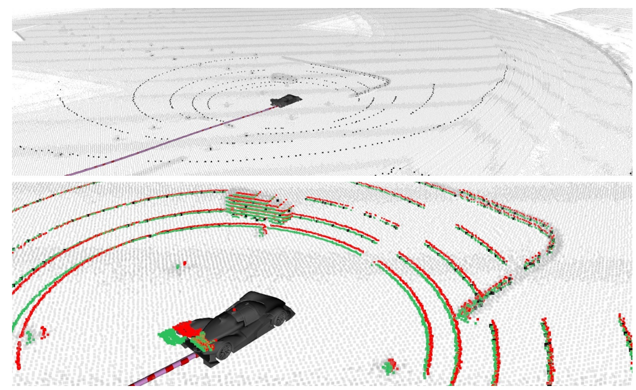

Fig. 7. Top: Far distance view on filtered point cloud (thick black points) used for localization. Bottom: Effect of lidar distortion and its correction, containing offline generated map (grey), lidar measurement (red), and corrected point cloud (green).

During the mapping process, only the **os1_node** is running and storing the data on the vehicle. Afterwards, the **ndt_mapping** node processes point cloud data offline, generating the 3D point cloud used for localization of the racetrack. NDT mapping does not require any additional sensors, and the **ndt_mapping** package is very simple to use. The input of the mapping module is solely the lidar measurement without any preprocessing. While the incoming scans are sequentially added to the map, previously seen regions are taken into account to stabilize the simultaneous localization and mapping (SLAM) procedure. The point cloud map is further reduced and filtered by a voxel grid filter. With this, the quality and usability of the map is increased significantly.

The algorithm for the mapping is re-used for the localization, with slightly different settings and a modified procedure, which increases the real time capability. The **ndt_matching** node receives distortion-corrected and down sampled measurement cloud as an input, and it outputs the position and statistical information about the overall matching quality. While the mapping approach uses the full point cloud, the online localization only takes a small fraction for this task. The NDT matching algorithm uses the created point cloud map provided by the **map_loader**. The point cloud map of the racetrack has a different coordinate system compared to the GPS coordinates system. Therefore, the map loader transforms the point cloud map to the GPS coordinate system. This step is used only if an evaluation of the lidar localization is necessary, otherwise the lidar localization can have its separate coordinate system.

To handle the amount of measured points, the point cloud is down sampled with the **points_downsampler**. There, a voxel grid filter samples down the data in a three-dimensional space. The input point cloud is separated into a set of cubes (voxels) with a defined edge length (voxel leaf size). All measured points in the voxel are reduced to one measurement, leading to a high data reduction, especially in the near field of the sensor.

Lidar sensor usually provides data in one single point cloud, which is provided at the end of a complete scan. All points stored in this point cloud, refer to the same origin (the sensor's position). Once the sensor is moving during a scan process, single measurements refer to globally different reference locations, leading to errors in distance and direction. These errors, caused by the so-called *lidar distortion*, increase with the speed and the turn rate. As the performance of localization is highly dependent on the accuracy of the measurements, distortion needs to be corrected. The **lidar_distortion** node corrects the distortion of the input point cloud, based on the approach given in [18]. Fig. 7 shows the measured lidar point cloud (red) and the corrected point cloud (green).

C. Time synchronization

To increase performance and accuracy of the lidar-based localization, it is important to know the exact measurement time. For example, when the race car drives with a velocity of 30 m/s, a time delay of 10 ms results in an error of about 0.3 m. Therefore, all time delays, starting from the measurement till the execution of the NDT algorithm, need to be considered. The lidar-based localization provides the position when the measurement was started. During the position calculation, the DevBot 2.0 continues moving and is at another position. This results in a position that is older than 40 ms. For the compensation of the time delay, the vehicle state and the estimated position of the NDT algorithm are used to predict the position at the current time. For the prediction, a single-track model is used, where input velocity and turn rate are provided by the vehicle odometry.

V. Experiments

The described approach was evaluated at two RR competitions, using the DevBot 2.0. The first event was located at the Zala Zone proving ground in Hungary and the second one at the Circuit de Croix-en-Ternois in France. For the localization performed on the two different tracks, the 360° scanning Ouster OS1 lidar with a horizontal resolution of 1024 points was used. While the mapping was done with the 64 layer OS1-64, the localization used the 16 layer OS1-16. Using less layers reduces the amount of points significantly, allowing to process point clouds faster, while still decent accuracy is provided. The sensor is mounted at a height of 1.1 m. Two GPS antennas, used for performance evaluation, are mounted at the same height in front and behind the lidar. As reference measurement system an Oxford RT4000 with an inertial and position measurement unit is used in the race car. The target hardware platform for the lidar localization and the other racing relevant software components is a NVIDIA Drive PX2 computing platform. While racing, the car reaches longitudinal and lateral accelerations above 10 m/s^2, as well as velocities above 45 m/s.

The results from the localization competition in Zala Zone are explained in detail, while the better weather conditions provided higher lateral accelerations and higher tire grip. This also makes the lidar localization more difficult, because of the higher changes in acceleration between consecutive point cloud measurements from the lidar. During the race, no GPS data was available. Therefore, the data from the training session was used to evaluate the localization accuracy. Fig. 8 shows a histogram of the error between the lidar-based localization and the reference GPS position. Most of the time the error is below 20 cm and is mainly caused by longitudinal driving direction, based on the timing issues in a ms range of the lidar measurement uncertainty. The lateral error of the lidar-based localization is less than 10 cm. A top-down view,

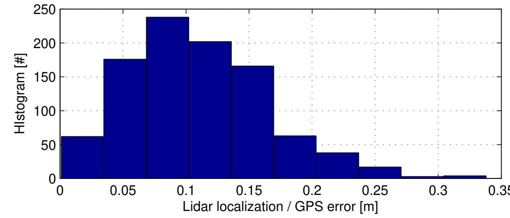

Fig. 8. Histogram of the position error between the lidar-based localization and the reference GPS position from the fastest lap at the training session.

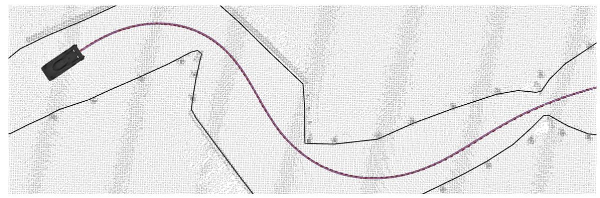

Fig. 9. Top-down view of curve number 8, 9, and 10 of the racetrack. Blue: GPS reference measurement. Red: lidar-based localization

given in Fig. 9, shows the history of the GPS (blue) and the lidar localization (red). Both lines are overlapping, no lateral differences to the reference GPS measurement is visible. The position error between the lidar-based localization and the GPS system is visualized over time in Fig. 10. The error increases in situations of high changes in both longitudinal and lateral acceleration of the autonomous race vehicle.

The Autoware's NDT implementation provides a reliability info, i.e., how good the result of localization is. The NDT reliability is visualized in Fig. 11. It depends on the number of iterations needed to find the position and the position estimation accuracy. Lower values correspond to faster convergence of the algorithm. It can be stated that in every curve, with high lateral accelerations, the algorithm needs more time to converge and the score of the NDT reliability increases. In order to analyze the maximum performance of the algorithm and to investigate if an ARM architecture is sufficient for intensive lidar localization, a CPU version of the NDT is used. Fig. 12 shows the CPU load of the NVIDIA Drive PX2 during the race. It has two parallel computing platforms (Tegra A/B), where each platform has three cores. All safety software components, path planning tasks, and the interface to the lidar are running on Tegra A. On Tegra B, the lidar localization and the data recording are executed. The average load on Tegra A does not reach critical limits: all CPU cores are in average below 70 %. However, the lidar localization causes high CPU load: often one of the core reaches 100 % load. Detailed analysis shown, this is caused by a significant change in velocity or turn rate. The difference between the last and the current position becomes greater, aggravating the determination of the current position by the NDT and thus increases the computation time. However, the underlying sensor fusion of lidar-based position and vehicle odometry can handle a jitter of the lidar-based localization.

VI. Conclusion

This work analyzed and evaluated entire process of lidar-based mapping and localization that was used for ROBORACE Season Alpha racing challenges. It explained overall

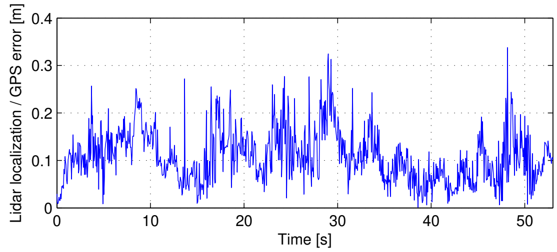

Fig. 10. Position error between the lidar-based localization and the reference GPS position from the fastest lap at the training session.

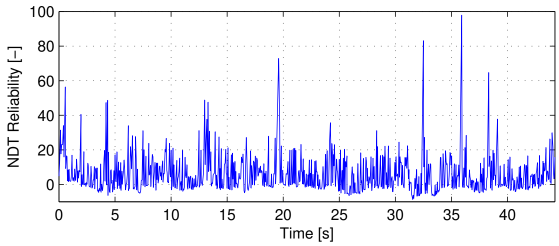

Fig. 11. NDT reliability during the race. A lower value represents a faster convergence. Spikes in the reliability are related to curves on the track.

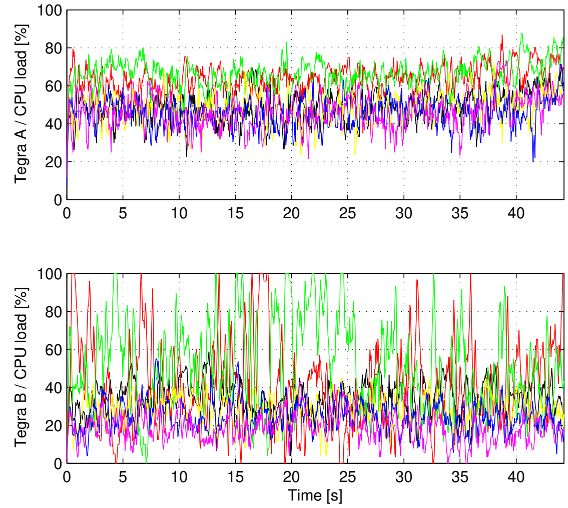

Fig. 12. CPU load during the race. Each color shows a different CPU core. Top: Tegra A used for safety components, path planning and lidar interface. Bottom: Tegra B used for logging and lidar localization.

procedure used on the racing events: from high precision mapping of a racetrack to the online localization by using only lidar sensor. Proposed structure is adaptive and possible to extend in a way to meet the challenging requirements of high speed dynamic driving, while it provides accurate, robust and low complexity lidar localization. The results were successfully validated with the DevBot 2.0 racing car in two racing challenges. Even without GPS sensor and with the limited performance of the computing platform, accurate and precise localization is possible to achieve.

ACKNOWLEDGMENT

The authors kindly acknowledge financial support of the COMET K2 – Competence Centers for Excellent Technologies Programme of the Federal Ministry for Transport, Innovation and Technology (bmvit), the Federal Ministry for Digital, Business and Enterprise (bmdw), the Austrian Research Promotion Agency (FFG), the Province of Styria and the Styrian Business Promotion Agency (SFG), and express their thanks to their partners, AVL List GmbH and to Infineon Technologies Austria AG for financial support. We express our gratitude to ROBORACE for the opportunity of evaluating our algorithms on their prototype and for their support during the testing sessions. We would like to thank the TUM team for their impressive contribution as well. Finally, we would like to thank all members of the team *Autonomous Racing Graz* for the hard, time-of-day independent work on and next to the racetrack.

REFERENCES

[1] On-Road Automated Driving (ORAD) committee, *J3016 Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*, Jun. 2018.

[2] A. Heilmeier, A. Wischnewski, L. Hermansdorfer, J. Betz, M. Lienkamp, and B. Lohmann, "Minimum curvature trajectory planning and control for an autonomous race car," *Vehicle System Dynamics*, vol. 0, no. 0, pp. 1–31, 2019.

[3] J. Zhang and S. Singh, "Loam: Lidar odometry and mapping in real-time.," in *Robotics: Science and Systems*, vol. 2, 2014.

[4] J. Levinson, M. Montemerlo, and S. Thrun, "Map-based precision vehicle localization in urban environments.," in *Robotics: science and systems*, vol. 4, 2007.

[5] R. W. Wolcott and R. M. Eustice, "Fast lidar localization using multiresolution gaussian mixture maps," in *2015 IEEE ICRA*, 2015.

[6] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: Part i," *IEEE robotics & automation magazine*, vol. 13, no. 2, pp. 99–110, 2006.

[7] *Global championship of driverless cars*, https://roborace.com/, [Online; accessed 31-March-2020], Roborace, 2016.

[8] T. Stahl, A. Wischnewski, J. Betz, and M. Lienkamp, "ROS-based localization of a race vehicle at high-speed using LIDAR," *E3S Web Conf.*, vol. 95, p. 04 002, 2019.

[9] J. Kabzan, M. de la Iglesia Valls, V. Reijgwart, H. F. C. Hendrikx, C. Ehmke, M. Prajapat, A. Bühler, N. B. Gosala, M. Gupta, R. Sivanesan, A. Dhall, E. Chisari, N. Karnchanachari, S. Brits, M. Dangel, I. Sa, R. Dubé, A. Gawel, M. Pfeiffer, A. Liniger, J. Lygeros, and R. Siegwart, "AMZ driverless: The full autonomous racing system," *CoRR*, vol. abs/1905.05150, 2019. arXiv: 1905.05150.

[10] N. Akai, L. Y. Morales, T. Yamaguchi, E. Takeuchi, Y. Yoshihara, H. Okuda, T. Suzuki, and Y. Ninomiya, "Autonomous driving based on accurate localization using multilayer LiDAR and dead reckoning," in *2017 IEEE 20th ITSC*, Oct. 2017.

[11] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda, "A survey of autonomous driving: Common practices and emerging technologies," *CoRR*, vol. abs/1906.05113, 2019. arXiv: 1906.05113.

[12] *3D point cloud and mesh processing software*, http://www.cloudcompare.org, [Online; accessed 27-March-2020], CloudCompare, 2012.

[13] P. Bergström and O. Edlund, "Robust registration of point sets using iteratively reweighted least squares," *Computational Optimization and Applications*, vol. 58, pp. 543–561, 2014.

[14] J. Zubača, M. Stolz, and D. Watzenig, "Smooth Reference Line Generation for a Race Track with Gates based on Defined Borders," in *2020 IEEE Intelligent Vehicles Symposium (IV)*.

[15] F. Braghin, F. Cheli, and S. Melzi, "Race driver model: Trajectory planning," in *III European Conference on Computational Mechanics*, 2006.

[16] P. Biber and W. Strasser, "The normal distributions transform: A new approach to laser scan matching," in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)*, vol. 3, 2003.

[17] S. Kato, S. Tokunaga, Y. Maruyama, S. Maeda, M. Hirabayashi, Y. Kitsukawa, A. Monrroy, T. Ando, Y. Fujii, and T. Azumi, "Autoware on board: Enabling autonomous vehicles with embedded systems," in *2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS)*, 2018.

[18] T. Renzler, M. Stolz, M. Schratter, and D. Watzenig, "Increased accuracy for fast moving LiDARS: Correction of distorted point clouds," in *2020 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*, 2020.