OTTO-VON-GUERICKE-UNIVERSITÄT MAGDEBURG

FACULTY OF COMPUTERSCIENCE
Institute for Intelligent Cooperative Systems

Master's Thesis

Autonomous Driving in Urban Centers - Roundabout Monitoring

Julian-B. Scholle
April 3, 2017

Otto-von-Guericke-Universität Magdeburg
Fakultät für Informatik
Universitätsplatz 2
39106 Magdeburg
Germany

Eidesstattliche Erklärung

Ich erkläre hiermit an Eides statt, dass ich die vorliegende Arbeit selbständig und ohne unerlaubte fremde Hilfe angefertigt, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt habe. Die aus fremden Quellen direkt oder indirekt übernommenen Stellen sind als solche kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form keinem anderen Prüfungsamt vorgelegt und auch nicht veröffentlicht.

Göteborg, den April 3, 2017

Julian-B. Scholle

Acknowledgements

I would especially like to thank Associate Professor Christian Berger and J. Professor Sebastian Zug for their organizational and technical support during and especially in the run-up to this work and the “Deutschen Akademischen Austauschdienst” - DAAD for their financial support during my stay in Gothenburg. I would also like to thank all the other colleagues from Gothenburg, who have supported me professionally and morally.

Index of Abbreviations

DBSCAN Density-Based Spatial Clustering of Applications with Noise

SVD Singular Value Decomposition

LLSQ linear least squares

RANSAC Random Sample Consensus

MKS multi-body simulation

ACC Adaptive Cruise Control

DARPA Defense Advanced Research Projects Agency

CTRV Constant Turn Rate and Velocity

Contents

1 Introduction

bezug auf kreisverkehre fehlt

Autonomous driving and the networking of vehicles with their environment are, together with electromobility, the most frequently discussed topics in the automotive sector. Rightly so: Autonomous driving has the potential to create completely new structures in the mobility market. [1]

So also, the Chalmers University of Technology, which has also initiated the project "CampusShuttle" in addition to Volvos' "DriveMe" project, is an interdisciplinary and cooperative research project at the Chalmers University of Technology and the University of Gothenburg. The project is located in the ReVeRe (Chalmers Research Vehicle Resource). The vision is a self-driving car between the two campuses of Chalmers.

Within the scope of the project, the vehicle is to be examined in various traffic scenarios. The focus is on urban transport and the vehicle must not only be able to interact with other cars, but also be safe with trams, buses, bicycles and all other traffic users.

1.1 Initial Situation

1.1.1 Test Platform

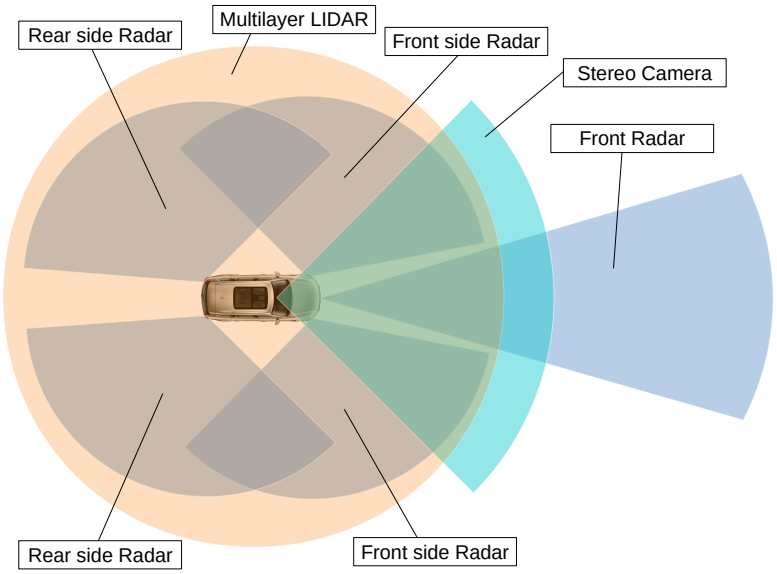
The test platform used in this work is a Volvo XC90 (2015) SUV, named Snowfox (see fig. 1.1). This test platform is equipped with many sensors for environmental monitoring. This includes five radar sensors, all around the vehicle, where front radar has a wider range. As well as a stereo camera and a Velodyne VLP-16 LiDAR. The arrangement of the sensors can be taken from fig. 1.2.

A Applanix POS LV is installed in the vehicle in addition to the environmental sensor system and the standard vehicle sensors (for example odometer, intertial sensors). At the time of writing this work, it was unfortunately not yet possible to access the radar sensors and the stereo camera. Therefore, only the Velodyne LiDAR and the Applanix system will be described in detail.

1. https://www2.deloitte.com/de/de/pages/consumer-industrial-products/articles/autonomes-fahren-in-deutschland.html (03/09/2017)

Figure 1.1: Test Platform Snowfox

Figure 1.2: Snowfox Sensors



Velodyne VLP-16 LiDAR

The Velodyne VLP-16 is a 360 degree 3D laser scanner with a rotational speed of 5 to 20 revolutions per second [40]. It provides a vertical FOV of 30 degrees, at 2 degrees resolution. With a range of 100m it can cover a circumference of 200m diameter. Furthermore, the VLP-16 can be syncronized with the Applanix POS LV, which allows a low-jitter signal. A further function of the Velodyne sensor is that it can react to different measuring pulses. By evaluating the last pulse instead of the strongest pulse it is possible to see through transparent objects. This allows us to determine the width of the vehicle in a later part of this work, as the Velodyne can look through the glass windows of the vehicle. At a set speed of 10Hz, the VLP-16 provides a resolution of 0.2 degrees with a variance of +- 3cm. The VLP-16 is centered on the roof of the XC90 in order to achieve the highest possible positioning to achieve a panoramic view of the vehicle. It is important to note that this alignment is unacceptable to the sensor because the sensor has a vertical field of view of

-15 to +15 degrees. As a result, all measurements over zero degrees are practically useless. The view of the manufacturer side [2] reveals that the VLP-16 has been constituted for use with drones, while the larger HDL64E [3] is advertised explicitly for the urban automotive sector, and has a field of view of +2 to -24.9 degrees and thus for use in the Automotive sector appears to be more appropriate. The resulting problems will be discussed later.

Applanix POS LV

The POS LV is a compact position and orientation system. It offers stable, reliable and reproducible positioning solutions for land-based vehicle applications. The POS LV provides an inertial sensor and odometry based position measurement with an accuracy of up to 0.3m (up to 0.035m when using the RTK correction). Furthermore, the heading delivered by the POS LV is also used, which provides an accuracy of 0.2 degrees. Even after losing the GPS signal, the POS-LV can provide a position through its odometer and the inertial sensor. However, this will deteriorate over time so that an accuracy of 2.51m can be expected 60 seconds after the GPS signal is lost. [14]

1.2 Research Goal

Since autonomous driving is a very wide, indisciplinary subject, it is obvious, that not everything can be dealt within this work. Within the scope of DARPA Chalenge many papers were published on this subject. What has not yet been explicitly discussed in these publications is the explicit handling of roundabouts, with autonomic vehicles, also the author is not aware of any further publications. So the aim of this thesis is therefore to analyze what sensor equipment is necessary for the observation of roundabouts, or whether the existing sensor equipment of the ReVeRe test vehicle Snowfox can be regarded as sufficient.

2. `http://velodyneLiDAR.com/vlp-16.html` (03/09/2017)
3. `http://velodyneLiDAR.com/hdl-64e.html` (03/09/2017)

2 Basic Knowledge

2.1 Roundabouts

reference

Roundabouts are growing in popularity in Germany. As seen in fig. 2.1, they have a smaller number of points of conflict (8) as opposed to crossroads (32), thus contributing greatly to road safety, which reduces the rate of accidents.

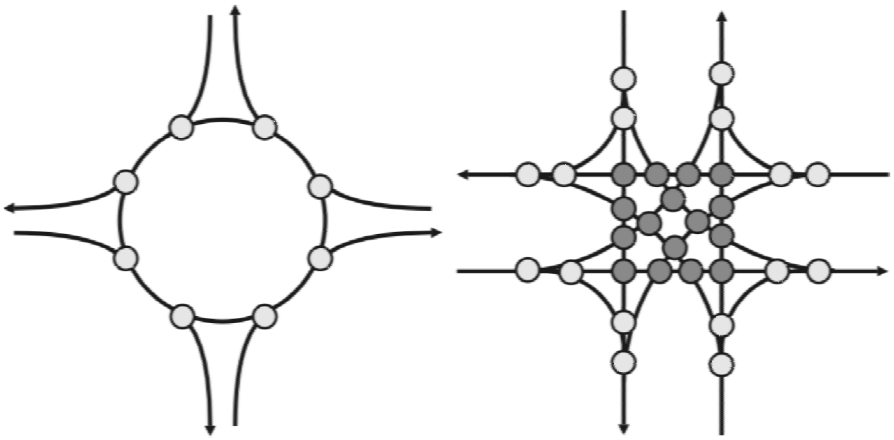
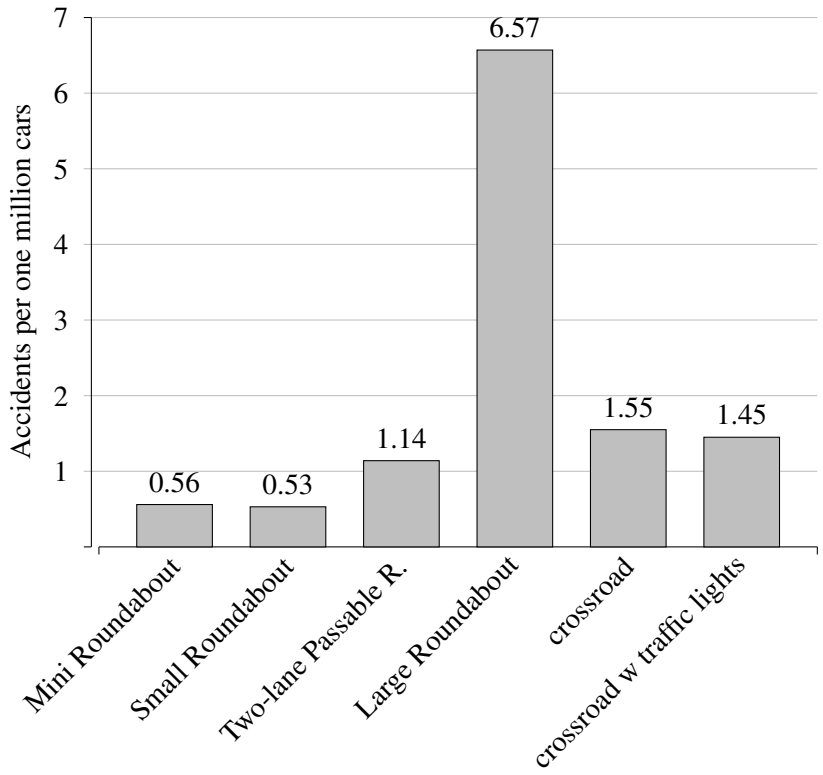


Figure 2.1: Roundabout Conflict Points [36]

In [4, 8, 9, 41], is examined, the safety of roundabouts in the inner city area. The evaluation was graphically prepared. Where fig. 2.2 shows the accident rate, which makes a statement about the general risk of an accident. While fig. 2.3 shows the accident charge rate, which divides the costs of the accidents on all vehicles and represents a measure of the severity of the accidents, both material and personal injury are included. It can be seen that the mini roundabouts [section 2.1.3] and small roundabouts [section 2.1.3] have a significant lower risk than other intersection points, in both the risk of accidents and severity. While large roundabouts are roughly comparable with normal crossings. In [7] this is reasonable to the higher speed in this roundabout. From this it can be deduced that the small and mini roundabouts from the point of risk, are to be preferred over the large roundabout. It should be noted that during the investigation of [7] with 100 roundabouts it is noticeable that 83% of the roundabouts have a diameter of less than 35m and can therefore be assigned the

class mini or small roundabout. Although this study is not representative, it is to be assumed that the number of small and mini roundabouts in the inner-city limits is larger than that of the large roundabouts, which seems reasonable on account of the space conditions in the inner-city limits. Furthermore, circular traffic also allows nodes to be constituted with more than 4 roads, which gives them a further advantage. Unfortunately, there are no publications known to the author in the context of autonomous driving and roundabouts.

Figure 2.2: Accident Rate in City Limits



Even if roundabouts for the human driver seem to be a relief, it is necessary to investigate the challenges in regard to autonomous driving. In the following, we present an overview of the legal aspects in Germany.

2.1.1 Roundabouts in Law

In Germany, there is no law stipulating the exact construction of roundabouts. Instead, the elements of the rural roads and city streets are dealt with in "Directives for the Design of rural roads" [37] and the "Directives for the Design of Urban Roads" [38]. These guidelines are also relevant to the choice of a convenient junction type when linking roads. The considerations discussed there are based on traffic variables, area-related characteristics, economic criteria and spatial planning or urban planning requirements. The guidelines also regulate the basic design and operational formation of roundabouts. The Directives for the Design of Urban Roads [38] are relevant for this dispute. Since the access the RASt ist limited, most of the information is coming from [36] whereupon RASt is based on.

2.1.2 Elements of a Roundabout

Definition 2.1 (roundabout island) *The roundabout island is the constructional area in the middle of the roundabout, which is surrounded by vehicles.*

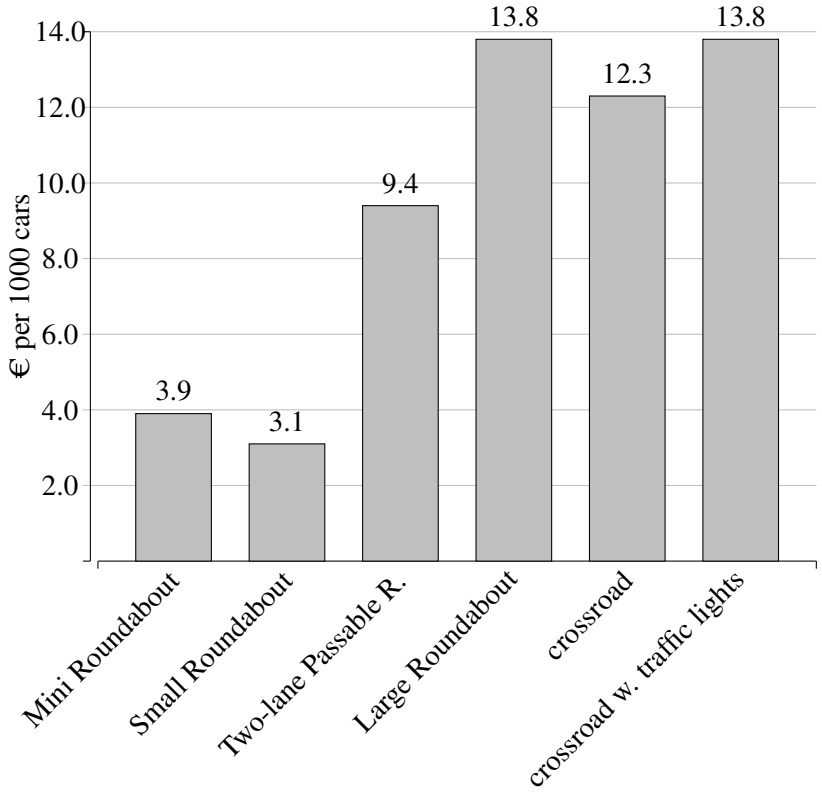


Figure 2.3: Accident Charge Rate in City Limits

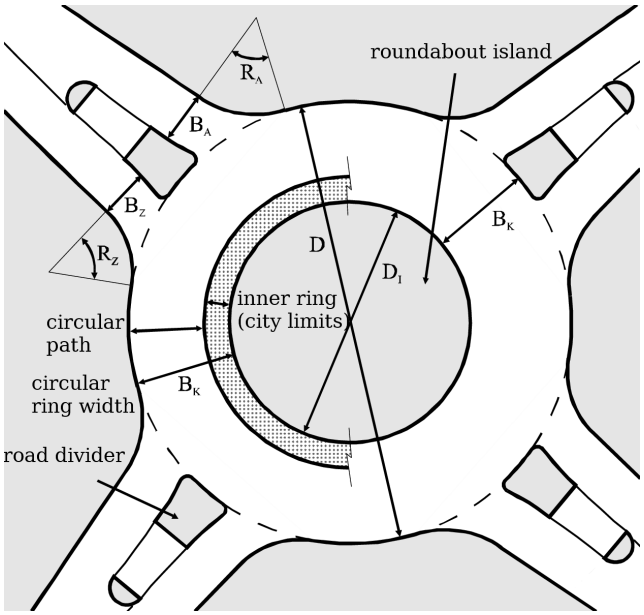


Figure 2.4: Definition of individual design elements and dimensions of a roundabout [36]

For miniature roundabouts, the roundabout island is crossable. [36]

Definition 2.2 (circular path) *The circular path is the road that serves to drive the roundabout island. An inner ring, if present, is not part of the circular path (VwV-StVO zu §9a V., Rn. 5). [36]*

Definition 2.3 (circular ring with (B_K)) *The structural width includes the circular track and a paved inner ring, if any. It is dependent on the outer diameter and the desired traffic routeing (one or two lanes). The edge strip width is oriented on the relevant continuous roadway. [36]*

Definition 2.4 (outer diameter (D)) *The outer diameter is measured at the outer edge of the circular ring. It is the essential measure for describing the size of the roundabout. [36]*

Definition 2.5 (inner diameter (D_I)) *The inner diameter is the diameter of the roundabout island. [36]*

Definition 2.6 (road divider) *The road divider is the structurally designed island between the circular exit and circular driveway. It serves to separate the circular exit and circular driveway, the management of the traffic, as well as the pedestrians and cyclists as cross-bordering aid. [36]*

Definition 2.7 (lane width of the circular driveway (B_Z) and circular exit (B_A)) *The width of the circular driveway and exit is measured at the beginning of the corner. [36]*

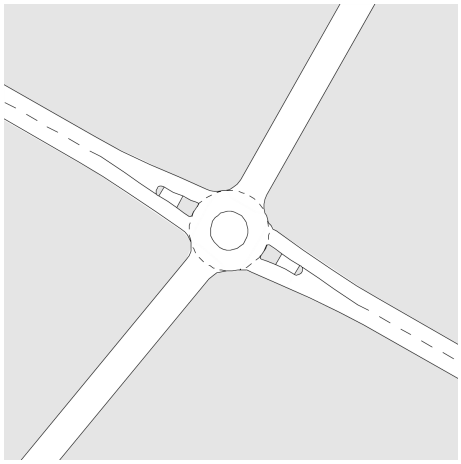
Definition 2.8 (Corner rounding radius (R_Z and R_A)) *This is the radius of the rounding at the right edge of the road between the circular driveway and the circular path. For a elliptical arch with a radius sequence of three different radii, R_Z is the radius R_2 of the central arc. When the road edge is formed as a tractrix, R_Z is the smallest radius of the road edge. [36]*

2.1.3 Types of Roundabouts

There are several types of roundabouts, which are differentiated by the different application criteria and the partly different design principles according to the situation inside and outside built areas. Furthermore, a division is made as a function of its size. [36]

Mini Roundabout

Figure 2.5: Mini Roundabout [36]



Within built-up areas, smaller outer diameters are possible under certain conditions. These roundabouts are called mini roundabout. The roundabout island must then be capable of being passed over. The outer diameter should be at least 13 m, so that the circular island does not become too small. Larger outer diameters make driving easier. Outer diameters of more than 22m, however, do not offer any transport advantages. From an outside diameter of about 22 m, therefore, the installation of a small roundabout with 26 m is generally more convenient. Bypasses are generally not required in the areas where mini roundabout can be used.[36]

Small Roundabout

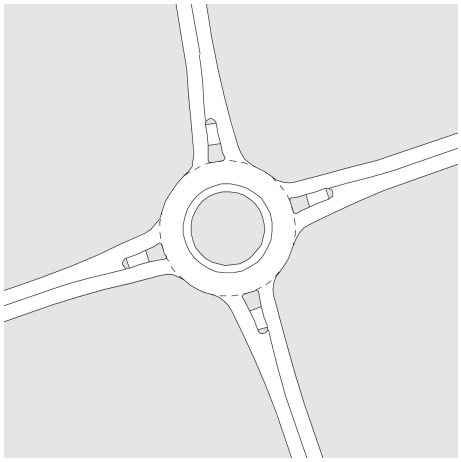


Figure 2.6: Small Roundabout [36]

The small roundabout has a single lane circular path and single lane circular driveways and exits. The roundabout island is not passable and can be building area. The outer diameter must be at least 26 m. Bypasses can be set up for driving geometric reasons or to increase performance.[36]

Two-lane Passable Roundabout

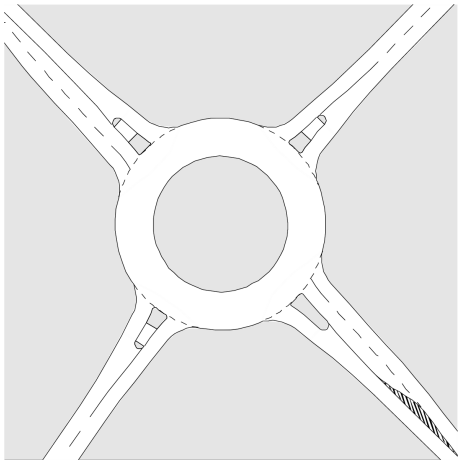


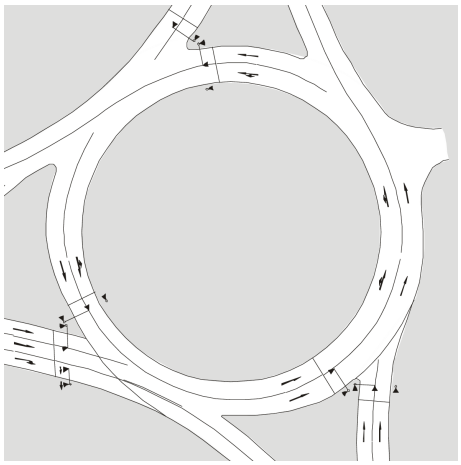
Figure 2.7: Two-lane Passable Roundabout [36]

If the capacity of the small roundabout is not sufficient and can not be ensured by the installation of bypasses, the circular path of a small roundabout can be designed to be two-lane driveable. At such a roundabout, the circular path is so wide that cars can travel side by side in a circle. If a further increase in the capacity is required, individual circular driveway can also be carried out in two lanes, if pedestrians and cyclists are not to be considered regularly. For safety reasons, circular exits are always carried out in single lanes. For geometrical reasons, the outer diameter must be at least 40 m for two-laned accessibility.[36]

Large Roundabout

Large Roundabouts with two or more lanes marked by markers on the circular path should be operated only with a light signaling system and closely coordinated roundabout design and traffic control.[36]

Figure 2.8: Large Roundabout [36]



2.2 Random Sample Consensus

The Random Sample Consensus (RANSAC) [16] is an algorithm for estimating a model within a series of readings with outliers and rough errors. Due to its robustness, it is mainly used in the evaluation of automatic measurements primarily in the field of image processing. In this case, RANSAC is supporting traditional compensation methods such as the least squares method, which usually fail with a larger number of outliers, by calculating an outliers free data set, the so-called consensus set.

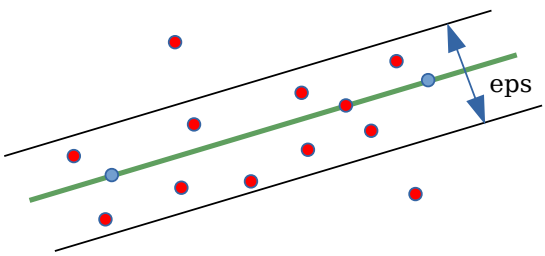
The RANSAC requires more data points than is required for the unambiguous determination of the model. From this set of data points randomly as many data points are selected, as necessary to define the model unambiguously. From the remaining data, those which have a spacing which is smaller than a certain limit value are then selected. This set now represents the “consensus set”. If it contains a certain minimum number of values, a good model was probably found and the consensus set is stored. These steps are repeated several times. Then the subset that contains the most points is selected. Using this subset, the model parameters are calculated using one of the usual compensation methods. The RANSAC therefore has three parameters to determine, which influence the result:

- number of iterations
- minimum size of the “Consensus Set”
- distance threshold value (eps)

Figure 2.9 shows an example for an good “consensus set”.

missing refernce to pics in dbscan and ransac

Figure 2.9: RANSAC

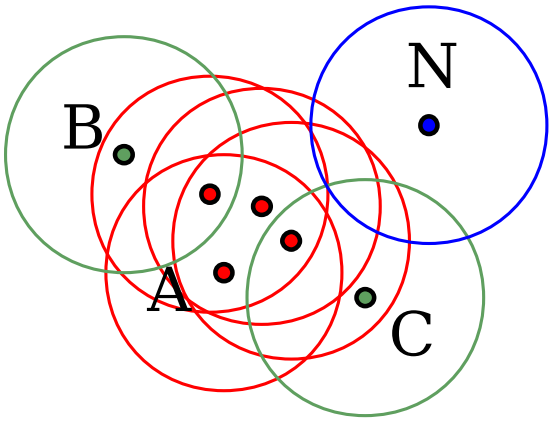


Note that the RANSAC is not a deterministic algorythm, beacuse of the random selection of the data points.

2.3 Density-Based Spatial Clustering of Applications with Noise

Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [15] is a deterministic data mining algorithm for cluster analysis. The algorithm is based on density connectivity, that means, it binds points based on its distance to clusters, DBSCAN, iterates over all data points that have not yet been processed, each processed data point is marked as processed. A range query request is then made for each of these points. If the size of the returned neighborhood is lower then a certain limit (minPts), the point is marked as noise. Otherwise, a new cluster is created by performing a new range request for each point in the neighborhood and adding the points to the cluster and marked them as processed. This is repeated until all the points in the cluster are marked as processed, so no further points can be reached in the neighborhood. Figure 2.10 shows an example, with a point threshold of one. So the blue point is marked as noise, because there are no neighbors inside of the maximum allowed distance. DBSCAN also makes a differentiation between core (red) and border (green) points, where core points are dense, means they can reach the minPts limit, while border points are not dense, but can be reached by point which is dense.

Figure 2.10: DBSCAN



The DBSCAN therefore has two parameters to determine, which influence the result.

- maximum distance of the neighborhood points
- minimum number of points required to form a dense region (minPts)

2.4 Middleware OpenDAVINCI

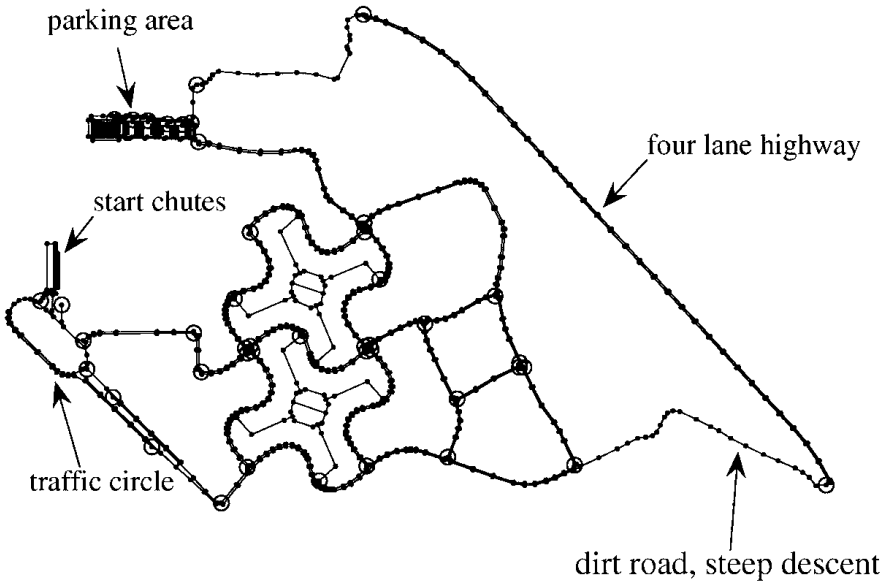
Autonomous software is typically a distributed system, on today's vehicles this system is based on ECUs and bus systems such as CAN, LIN. Distributed software makes it easier to integrate complex components within the system. In the area of autonomous driving, however, the historical structure of vehicles with ECUs and CAN is not optimal [10]. In order to handle the many components required, it is also advantageous to decouple components within an ECU or a

computing unit. For this purpose, there are several middelwares which handle and abstract communication within the components. As part of the Copplar project, the OpenDaVINCI middleware is used here. OpenDaVINCI is a real-time runtime environment designed for autonomous vehicles. OpenDaVINCI is based on Hesperia [6]. Communication between the components in OpenDaVINCI is based on UDP Multicast, which enables real-time communication between components and computing units [23] . For the communication, OpenDaVINCI offers time-triggered transmitters and data-triggered receivers, from which the data-triggered receiver is used to connect our software. Furthermore, OpenDaVINCI offers many further functionalities, which can be used, for example the handling of World Geodetic System 1984 (WGS84), for the conversion of GPS coordinates into local Cartesian coordinates. For this purpose, a reference GPS position is required, which should not be too far away to keep the calculation error small.

3 State of the Art

Even if autonomous driving is an actual research subject, there are so far no publications known to the author, which deal explicitly with roundabouts. In order to have a comparison to the sensor equipment and traffic monitoring, some papers of the DARPA Urban Challenge are used here. The DARPA Urban Challenge is a race between autonomous vehicles, which was launched in 2007 by the Defense Advanced Research Projects Agency (DARPA) [11]. The race took place in the built-up area of an abandoned barracks of the former Air Force base George Air Force Base, the map can be seen in fig. 3.1 [1].

Figure 3.1: DARPA Urban Challenge Map



The lanes were partially marked by lines, partially limited by concrete walls. All self driving cars were on the line at the same time. On one-, two- and four-lane streets, roundabouts, 4-way intersections with stop signs, blocked lanes and threading situations had to be successfully managed.

The evaluation is mainly limited to the final teams of the competition, since

1. http://rsta.royalsocietypublishing.org/content/368/1928/4649

they have most likely developed a functioning concept. The type and number of the sensors, for the object detection, used by the teams and the algorythms used for segmentation and tracking, if indicated, are evaluated.

3.1 DARPA Urban Teams

Team VictorTango

Team VictorTango [3] uses two Ibeo Alasca XT sensors, with a range of 200m and an opening angle of 240 degrees to cover the front area of the vehicle, the two sensors are complementary fused to increase the vertical resolution. Each sensor has a vertical resolution of four layers. For the rear area, a single 2D Ibeo A0 LiDAR is used. In addition, four Sick LMS291 LiDARs are used, but these are not used for object detection but are installed vertically in order to search for edges on the road in high resolution. However, the object segmentation is not implemented by Team VictorTango itself. In the meantime, a comercial ECU supplied by Ibeo is used, which directly detects objects in the data from the Ibeo sensors. The objects already finished are then filtered using the Road Detection and are then combined by a software component (Classification Center). And within that counter-checked against image processing with two 2D cameras. A classification is carried out only between moving and non moving objects, ie, all moving objects are cars. The team reached the third place in the competition [31].

Stanford Racing Team

The Stanford Racing Team [27] has a large number of sensors with the large Velodyne HDL-64E 3D LiDAR with a vertical resolution of 64 layers, a 360 degree panoramic view and a range of 120m. This is additionally supported by two SICK LD-LRS 2D LiDARs with 250m range at the rear of the vehicle, 2x Ibeo Alasca XT sensors in the front area and additional five Bosch long range radars at the front of the vehicle. To evaluate the measurement data, all LiDAR sensors are transformed into a virtual 2D laser scanner. The team describes, with the Velodyne sensor, problems with the roll and pitch angles of the vehicle which lead to false positives during the segmentation. These were corrected by the measurements of the Velodyne forming a round circle (see fig. 3.2). These circles have a fixed distance from each other, if the alignment is correct. The roll and pitch angles of the vehicle can be corrected by looking at these distances. In the case of the 2D sensors, the problem was limited by limiting the range.

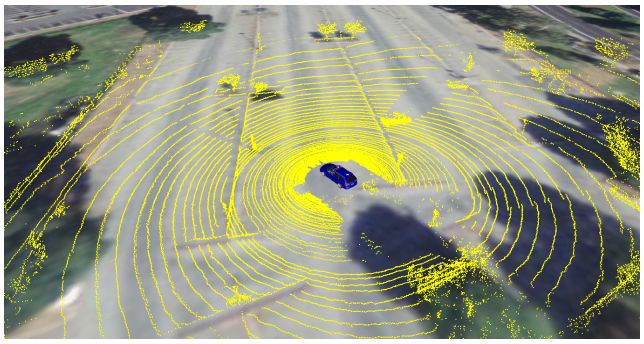


Figure 3.2: Velodyne HDL-64E Scan Rings [27]

For the segmentation of the objects, two time steps of the virtual laser scan are compared with one another. Areas in which a change is detected are then classified as objects and tracked using a particle filter. The tracker estimates, the position, the rotation, the speed, and the size of the object. The team reached the second prize in the competition [31].

Team Cornell

Team Cornell [12] uses two side Ibeo Alasca XTs for the object detection. In addition, three RADAR sensors are used on the vehicle front, which allow the Doppler shift to detect moving objects. The team has also evaluated a Velodyne 3D LiDAR, but decided on the price and the roll, pitch problems for the Alasca XT, which ejects four nearly parallel scans and makes it so easy to detect the ground. The team does not give any information about the segmentation of the objects, it is merely said that the sensor raw data is written into a local map, ie fused. For the actual tracking a "Rao-Blackwellized Particle Filter" [26] is used.

TerraMax

Team TerraMax [13] also uses the Ibeo Alasca XT and additional three SICK LMS-291 LiDAR sensors. The Ibeo sensor is front-mounted and the SICK sensors cover the lateral or rear area. Furthermore, the team makes intensive use of a self-developed panorama camera system, which is mounted in an identical version, once on the front and on the rear of the car. The system consists of three cameras each, depending on the speed, two cameras are used to calculate a stereo image of the environment. At higher speeds, the respective outer cameras are switched over in order to obtain a higher depth of view. In order to detect objects, this system is mainly used. In this case, a flat road is supposed, and everything which is not flat is detected as an object and then fused with the raw data of the LiDAR sensors. Team TerraMax does not describe any further tracking of the objects.

Tartan Racing

Tartan Racing [39] uses a combination of several sensors for environmental awareness, the Velodyne HDL-64E, the Ibeo Alasca XT, an ARS300 RADAR and various SICK 2D LiDARs. The segmentation of the objects is performed individually on each sensor and the detected objects are fused only in the next step. Unfortunately, the team does not make any precise statements about segmentation. Static objects are classified using a cost map and dynamic objects are tracked using an Extended Kalman filter and a "Constant Turn Rate and Acceleration" (CTRA) model and written into an object database. The team reached the first place in the competition [31].

Team Caltech

Team Caltech [2] use four Sick LMS 221, with a range of 30m but a high frequency of 75Hz, on each side of the vehicle. Since the team only uses 2D laser scanners, objects are simply segmented by uniting neighbored measurements without interruption. After this, the values are transformed into a local coordinate system. Due to the 75 Hz update rate of the sensors and the relatively slow speed of the vehicles in the race, a good linkage of new data to previously viewed objects can easily be made over a distance criterion. Then, the object

is passed to a Kalman filter which is updated to estimate position and velocity. If, after a certain number of observations, the estimated velocity of the object is above a threshold, it is classified as a car.

3.2 Conclusion

As we have seen, the teams share very different strategies to detect other vehicles. Some teams perform the segmentation before the fusing of the data, others afterwards. Some teams used a combination of image processing and LiDAR sensors, others trusted solely on LiDAR sensors. It is also noticeable that the two best teams (Tartan Racing, Stanford Racing Team) use a Velodyne HDL-64E, which has the highest resolution in comparison to all other sensors. Nevertheless, this is always combined with other sensors, which have a higher range. Presumably the sensor equipment had a great influence on the outcome of the competition. Comparing the test vehicles of the teams with our test vehicle "Snowfox" the sensor equipment is most similar to the teams which have the Velodyne HDL-64E, which is similar to the Velodyne VLP-16. However, this sensor used by "Snowfox" was not yet on the market at the time of the competition [2]. However, the VLP-16 has a significantly lower vertical resolution than the HDL-64E, but it is still above all other sensors. However, it must be said that the requirements in competition were below those of this project. Thus it was not necessary in the competition to distinguish between different traffic, since only the vehicles of the other teams were on the course. As other traffic, such as pedestrians and cyclists, have to be detected as part of this project, this places higher demands on classification and segmentation. Since pedestrians and cyclists are considerably smaller than vehicles, a high resolution is desirable. Whether the sensor is suitable for its task is to be investigated in the following.

2. http://www.spar3d.com/news/LiDAR/vol12no37-velodyne-announces-puck-LiDAR-sensor/ (03/23/2017)

4 Methodology

In the previous chapters we looked at the types of roundabouts and their components. We also reviewed the available test platform “Snowfox” and its sensor technology. We have found that the detection of objects in other projects often combines several and more expensive sensors in order to ensure a reliable detection of other traffic users or extend range.

We have determined that we want to check whether the sensor equipment of the test vehicle “ Snowfox ”, especially the Velodyne VLP-16, is suitable for the handling of a complex traffic scenario, the traffic monitoring of a roundabout.

For this purpose, an algorithm for detecting and tracking objects with the help of the Velodyne VLP-16 is proposed and implemented. The difficulty lies in the use of a single and in a comparatively low priced environmental sensor, which obviously has not been developed as a standalone solution for this purpose.

In its current application, this sensor offers a comparatively low resolution in the area wich is relevant for this work. Therefore, with many gradient-based algorythms, segmentation will often fail, beacuse the gradients becomes to small. For this reason, a ground plane based algorithm is implemented for the segmentation.

In addition, it is necessary to follow vehicles beyond the measuring horizon in roundabouts, with built-up central islands and multi-lane roundabouts, in order to ensure a safe entry into the roundabout. For this purpose, a tracking and state estimation algorithm is developed in section 6.3, which should grant that.

In order to evaluate the sensor setup with these algorythms, were collected several data collections at the Swedish AstaZero test area nearby Sandhult (see fig. 4.1 [1]). Some other experiments not carried out there, are carried out in a simulation made for this purpose. In this simulation a roundabout in a urban area with pavement and bikeway is designed, which the roundabout on AstaZero can not offer.

The evaluation is carried out manually by using of the graphically prepared measurement data. While the evaluation especially False-Negative and False-

1. http://www.astazero.com/wp-content/uploads/2016/09/%C3%96versiktsskiss_mod.pdf (03/24/2017)

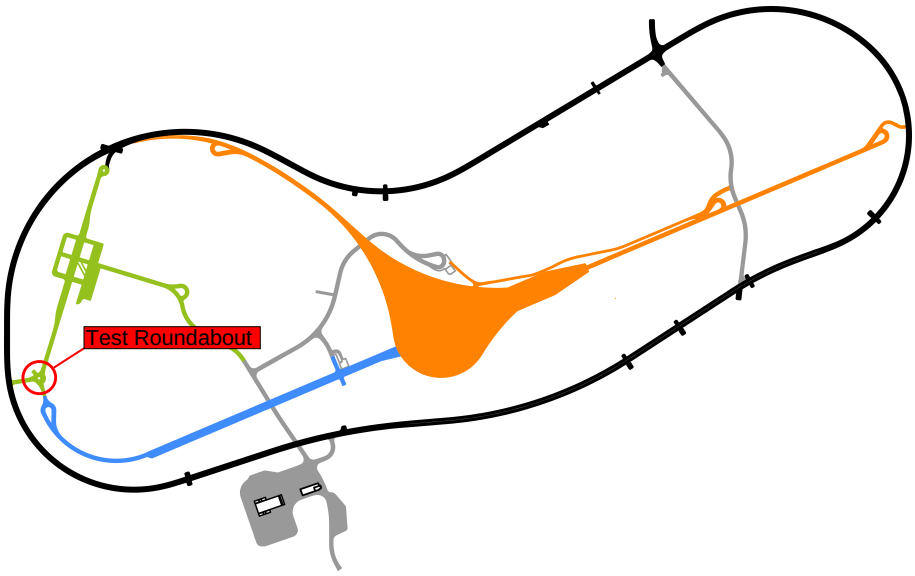


Figure 4.1: AstaZero Proving Ground

Positive detected obstacles will be discussed. Coarse outliers in the position or orientation of objects are also noted. Also, a performance analysis is made.

In order to evaluate the operability of the roundabout, a state machine is implemented which is intended to move the vehicle safely and accident-free through the roundabout. For this purpose, the simulation is monitored over a longer period of time and the number of possible collisions is noted.

5 Sensor Analysis

As already discussed, the selection of sensors will be limited to the Applanix POS-LV and the Velodyne VLP-16. Since the Velodyne is the more important sensor for the recognition of other traffic users, it is now investigated how far the sensor is suitable for the detection.

5.1 Theoretical Analysis

Under section 2.1.3 we have listed four different types of roundabouts. In order to evaluate the practical use of the Velodyne VLP-16 is now analyzed, how far the Velodyne is able to overlook the situation. For this purpose we will look at the size ratios in fig. 5.1.

Figure 5.1: Roundabout Sizes

Roundabout Type	min Size	max Size
Mini Roundabout	13m	22m
Small Roundabout	26m	40m
Two-lane Passable Roundabout	40m	-
Large Roundabout	>40m	-

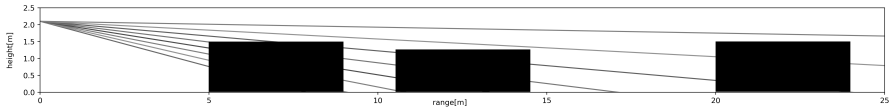
For a comparison to the Velodyne VLP-16, we will look at the visual range of the sensor in the 2D side view. The sensor provides 16 measurements at the same azimuth angle, with a distance of 2 degrees. Since for the evaluation, and unfortunately also for the detection, the angles greater than 0 degrees are not relevant, because they look into the air. So we only see the eight angles in the range of -1 to -15 degrees.

In this evaluation we assume that we try to detect a small car with a length of 4 meters and 1.5 meters height. In doing so, we start from the ideal situation that the vehicle drives in the direction of the test vehicle. Furthermore, we assume a mounting height of the sensor of 2.1 meters as this is finally the mounting height of the sensor on the test vehicle.

Range 0-25m For the range from 0m to 25m, we see in fig. 5.2 that in the worst case, we still get at least one measurement from the sensor. Is the vehicle

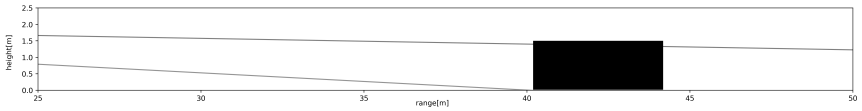
very much at the sensor, we get even a large part of the available resolution, whereby close to behind lying flat vehicles can be hidden. Therefore, we can see that the mini roundabout can be covered with the sensor, but we have to think about hidden obstacles.

Figure 5.2: Velodyne VLP-16 - Range 0-25m



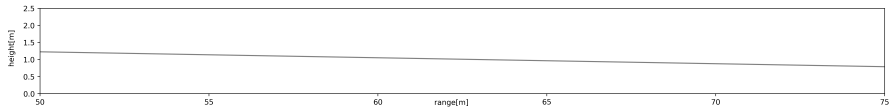
Range 25-50m For the range of 25m to 50m we see in fig. 5.3 that we can only get two measurements. For our assumed small car with a height of 1.5 meters, however, we do not keep a dead zone. However, for most measurements, we get only one measurement point and thus a very low resolution. Therefore the detection of vehicles in a small roundabout is already a challenge for the segmentation of the vehicles

Figure 5.3: Velodyne VLP-16 - Range 25-50m



Range >50m For the area greater than 50m, we see in fig. 5.4 that the resolution remains the same in the same way as for closer objects. However, there is a higher likelihood that vehicles have already been hidden in the previous area.

Figure 5.4: Velodyne VLP-16 - Range 50-75m



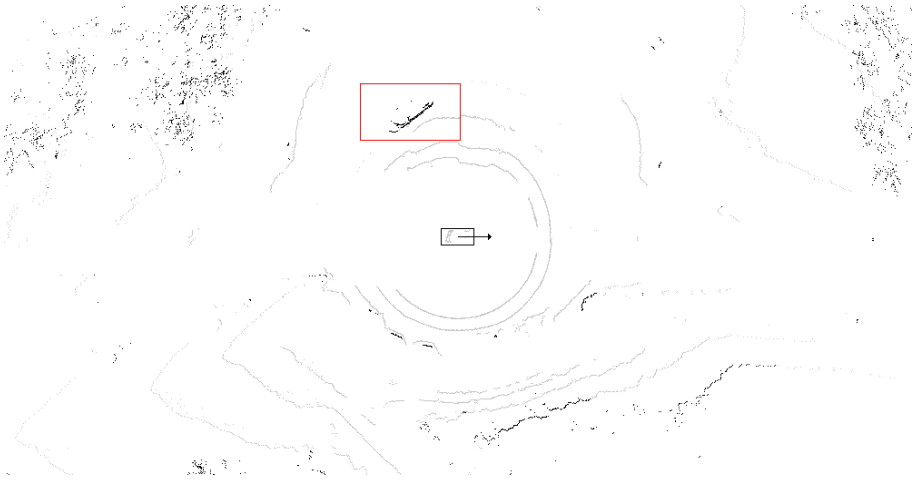
As we can see, we can excellent cover the area up to 25 meters and thus observe the mini roundabout completely, for larger roundabouts the sensor is only conditionally suitable. For all the following considerations, we shall consider only mini roundabouts or smaller small roundabouts. For all other roundabouts, evaluate whether it is necessary to observe all the vehicles in the roundabout, as the distance from driveway to driveway increases accordingly. It is maybe possible to consider the roundabout as a simple yield sign situation.

5.2 Practical Analysis

Since we have already looked at the theoretical considerations, we look at the practical measurements of the sensor. To do this, we can see a measurement on the test roundabout on AstaZero in the bird's eye view in fig. 5.5.

It can be seen here that the measurements in the front and especially in the rear region have gaps. Unfortunately, this can not prevented during installation of the Velodyne since the holding pre-treatment of the sensor does not provide a further height adjustment and no invasive changes should be made to the test

Figure 5.5: Velodyne VLP-16 - Hidden Layers

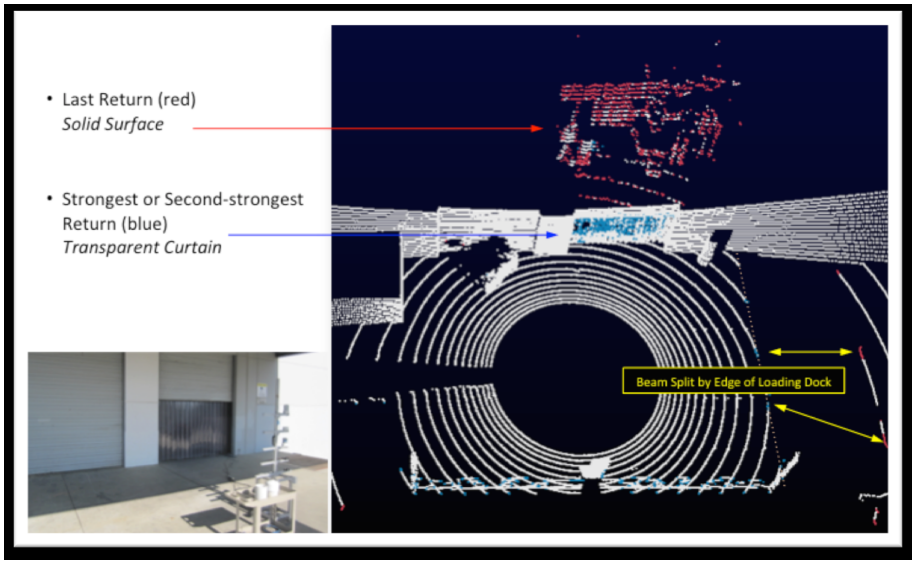


vehicle. Therefore, a compromise had to be concluded for the assembly, either to lose measurements in the front area, or to lose in the rear. The missing measurements are obscured by the vehicle roof. The smaller gaps in the front area come from the GPS antenna of the Applanix system and unfortunately also could not be prevented.

Since the monitoring of the rear area is not important for the roundabout observation, the missing measurements in the rear area are negligible. Since we have the highest resolution in the front area of the sensor, the missing measurements in the front range are not important, also these are only in the first measurement layer.

Further in fig. 5.5 to see is another test vehicle (red box). It can be seen from this that we also get measurements in the rear of the vehicle, which is actually hidden. This is explained by the fact that the sensor has several return modes, which allows it to see through transparent objects. This can be seen in the example in fig. 5.6.

Figure 5.6: Velodyne VLP-16 - Return Modes [40]



The sensor can be configured to return both measurements (Strongest Return and Last Return). Since the vehicles to be observed are not completely transparent and thus always provide sufficient measurements in the front area, in addition to keeping the data rate small, VLP-16 was configured to provide

only the Last Return.

6 Objekt Detection

In the following chapter, we will present the Velodyne VLP-16 with the detection of traffic users. We begin with the segmentation, that is, in the first step all measured values which do not represent potential traffic are removed. In the next step, the remaining measured values are combined with the aid of a clustering algorithm. In the third step the objects are abstracted and tracked, that is, we try to establish a relationship between the objects between different time steps. This is necessary to determine time-dependent variables such as the speed and rotation rate of the detected objects. In the last step, the objects are classified into types using the predetermined parameters.

6.1 Ground Removal

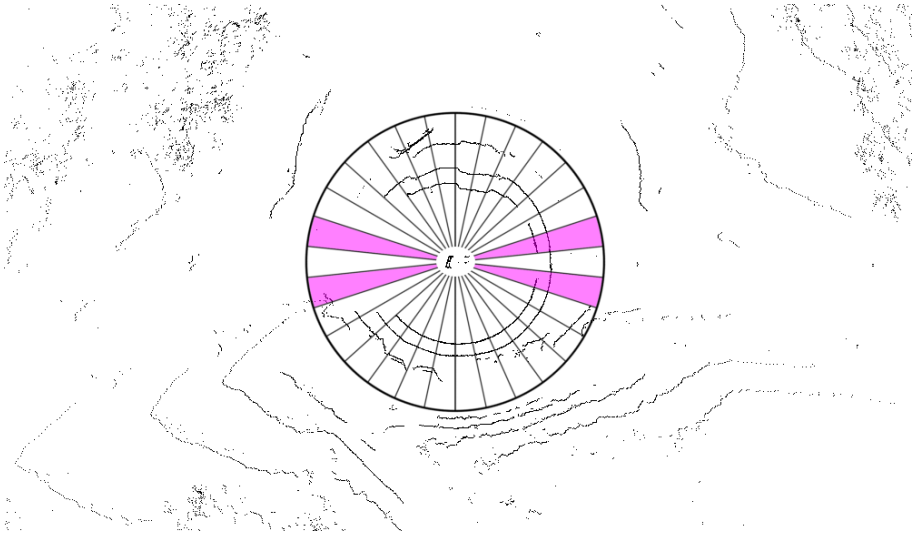
In order to recognize objects in a PointCloud, it is necessary to know which measurements belong to the ground and which belong to objects. There are many ways to achieve this. The most naive method is to remove the bottom plate by its z-coordinate. However, this method has many drawbacks, on the one hand, the LiDAR sensor has to be mounted exactly straight on a vehicle, on the other hand the vehicle must have a very stiff chassis in order to prevent a possible tilt of the sensor, as seen in the DARPA Challenge. Furthermore, this only allows the removal of planar surfaces, means only flat non-hilly grounds. Another common method is the removal of the base plate on the basis of a statistical mean value [42]. However, this method also requires a calibration of the sensor distance to the ground. And the determination of further threshold values which are dependent on the environment. The advantages of both methods are their low computational performance and runtime $\mathcal{O}(n)$. Better methods such as gradient-based expansion algorithms require a starting point that can be identified as a bottom plate. A further possibility is the description of objects as convex objects [28], which can also be described on the basis of the gradients. The advantage of this method is that no initial position is required for the bottom plate, but this is very processing power intensive.

For our application with the Velodyne VLP-16, the problem is in addition that the resolution of the sensor is very low in height. Depending on the distance of the vehicle within the required range, only two layers fall on the test vehicles, which means that gradient-based methods fail reliably, beacuse the gradients

are too small and the depletion of the necessary thresholds leads to frequent false positives. The method of the statistical mean and the method based on the z-coordinate, suffer from the chassis of the Volvo XC90 SUV. The height of the vehicle varies by several centimeters by changing the driving profile (Sport / Eco, etc.). Even slightly increased speed in the roundabout (about 30 km/h) lead to a clear lateral inclination of the vehicle. Therefore, another method is proposed. The detection of a ground plane in the measurement data.

For the detection of the ground plane we assume the following assumptions: the road can be presented approximately as a plane in $\mathbb{R}^3$. Further, the ground plane is the lowest plane in the mesurement area. Therefore, in the first step, the data set in polar coordinates is divided into 30 of cake piece formed segments. From this segments, two segments [fig. 6.1] are then selected in front and rear, which are not neighbored. The selection of the segments follows from the assumption that the road is in front or behind the vehicle. In future, the selection of the segments could also be optimized with the help of the vehicle steering angle or the valid range can be provided by a lane detection.

Figure 6.1: LiDAR Segments



Within these segments a search for the 10 measurements with the lowest Z value is made. The search is limited to the three lowest layers (-15, -13 and -11 degrees), since the measurements of all higher layers are too far away. The division into segments is therefore necessary in order to prevent all measured values running into a single local minima.

From this pre-filtered readings three are now randomly selected for a modified RANSAC, wich will run iteratively.

From these three points a plane is now formed in the Hesse normal form, which allows an efficient distance calculation to other points. After that, we collect all other points from our set of minima, using a distance criterion. This threshold is chosen as 0.2m experimentally. If can find more than 10 other points in our minima set, which fits our criteria, a new plane and their error is calculated from the plane and the new collected points by a plane fitting Algorithms [section 6.1.1]. The error is calculated over the sum of the squared distances of all points to the plane.

However, before we add the plane as a possible solution candidate, it is checked whether the layer is within a plausible parameter range. This means that the distance of the plane should move between 1.9m and 2.2m, which corresponds

approximately to the mounting height of the Velodyne sensor.

The number of iterations of the RANSAC is limited to 50. After the run of the RANSAC, the plane with the lowest error is taken and all points in the pointcloud are marked by their distance to the plane as ground. As distance threshold, an optimum value of 0.5 m was determined experimentally.

6.1.1 Plane Fitting

Macht das überhaut sinn? Laut ransac sollte das Planefitting nicht iterativ ausgeführt werden???? Fuck!

For plane fitting usually a Singular Value Decomposition (SVD) is used [29, 30, 35]. SVD has a complexity of $\mathcal{O}(\min\{mn^2, m^2n\}$ [20], since the plane fitting inside of the RANSAC is very often called with a large number of points, running to the SVD within the RANSAC causes a very high running time. For this reason, a linear least squares (LLSQ) algorithm with some optimization is used here. When using the LLSQ, it is important to note that the LLSQ is not optimizing the distance of the points to the plane but the distance of the points along an axis (in our case the z-axis), see fig. 6.2[1]. This can lead to problems, if the points are scattered far apart from the optimal plane. However, since we are using our preselected points from the RANSAC distance criterion, this poses no problem.

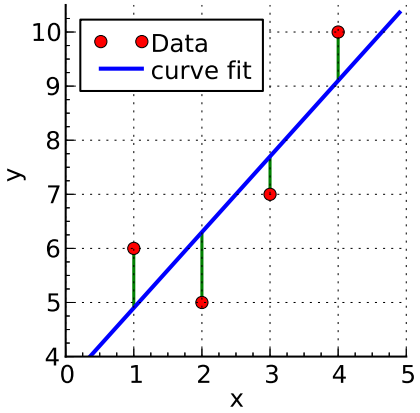


Figure 6.2: Linear Least Squares (LLSQ)

The representation of a plane in coordinate form is as follows: $a\vec{x}+b\vec{y}+c\vec{z}+d=0$. Since we consider a plane in $\mathbb{R}^3$, this system of equations is overdetermined. Since we want to optimize our plane in the direction of the z-axis, we set parameter c to 1 and can now simply solve our equation system by z: $a\vec{x}+b\vec{y}+d=-\vec{z}$. The vectors $\vec{x},\vec{y},\vec{z}$ represent the points to be taped. In matrix notation:

$$X\vec{\beta}=\vec{z}$$

$$\begin{bmatrix} x_0 & y_0 & 1 \\ x_1 & y_1 & 1 \\ & \cdots & \\ x_n & y_n & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ d \end{bmatrix} = \begin{bmatrix} -z_0 \\ -z_1 \\ \cdots \\ -z_n \end{bmatrix}$$

This system usually doesn't have a solution, but our real goal isn't finding an extact solution for $\vec{\beta}$, we want to find a good approximation $\hat{\beta}$ for this:

1. https://en.wikipedia.org/wiki/Linear_least_squares_(mathematics) (03/06/2017)

$$\hat{\beta} = \min(||\vec{z} - X\vec{\beta}||^2)$$

We can do this by multiplying our equation by the transpose of our point matrix X [18]:

$$(X^T X)\hat{\beta} = X^T \vec{z}$$

$$\begin{bmatrix} x_0 & x_1 & \dots & x_n \\ y_0 & y_1 & \dots & y_n \\ 1 & 1 & \dots & 1 \end{bmatrix} \begin{bmatrix} x_0 & y_0 & 1 \\ x_1 & y_1 & 1 \\ & \dots & \\ x_n & y_n & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ d \end{bmatrix} = \begin{bmatrix} x_0 & x_1 & \dots & x_n \\ y_0 & y_1 & \dots & y_n \\ 1 & 1 & \dots & 1 \end{bmatrix} \begin{bmatrix} -z_0 \\ -z_1 \\ \dots \\ -z_n \end{bmatrix}$$

This equation system can now be solved with the inverse of $(X^T X)$. Since the calculation of inverse matrices with $\mathcal{O}(dim^3)$ is also expensive, now another trick to save computing power. After multiplying with the transpose we get:

$$\begin{bmatrix} \sum x_i x_i & \sum x_i y_i & \sum x_i \\ \sum y_i x_i & \sum y_i y_i & \sum y_i \\ \sum x_i & \sum y_i & N \end{bmatrix} \begin{bmatrix} a \\ b \\ d \end{bmatrix} = \begin{bmatrix} \sum x_i z_i \\ \sum y_i z_I \\ \sum z_i \end{bmatrix}$$

The sums in the boundary areas of the matrix X and the vector $\vec{z}$ are good to see. We can set these to zero if we define all points relative to the mean point of all points, ie $P_i = P_i - \overline{P}$. Now we get:

$$\begin{bmatrix} \sum x_i x_i & \sum x_i y_i & 0 \\ \sum y_i x_i & \sum y_i y_i & 0 \\ 0 & 0 & N \end{bmatrix} \begin{bmatrix} a \\ b \\ d \end{bmatrix} = \begin{bmatrix} \sum x_i z_i \\ \sum y_i z_I \\ 0 \end{bmatrix}$$

Now we can also set d to zero, because if all our points are relative to the mean point, then our plane always runs through this point. Therefore we now can get rid of a complete dimension:

$$\begin{bmatrix} \sum x_i x_i & \sum x_i y_i \\ \sum y_i x_i & \sum y_i y_i \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} \sum x_i z_i \\ \sum y_i z_i \end{bmatrix}$$

The equation system can now be solved with the Cramer's rule

$$\begin{aligned} D &= \sum x_i x_i \cdot \sum y_i y_i - \sum x_i y_i \cdot \sum x_i y_i \\ a &= \frac{\sum y_i z_i \cdot \sum x_i y_i - \sum x_i z_i \cdot \sum y_i y_i}{D} \\ b &= \frac{\sum x_i y_i \cdot \sum x_i z_i - \sum x_i x_i \cdot \sum y_i z_i}{D} \\ \vec{n} &= [a, b, 1]^T \end{aligned}$$

It should be noted that the determinant can not be zero or near zero. However, since the angle between the vehicle and the plane is always close to 90 degrees, the determinant is typically very large. If the determinant is nevertheless close to zero (not equal to zero), the calculation is carried out in spite of

the fact that this also leads to a large error in the fitting. This is desirable at this point since the RANSAC sorts out invalid layers based on the error. If the determinant is exactly zero, the calculation is continued with a small value for the determinant.

komplexität und argumentation überprüfen

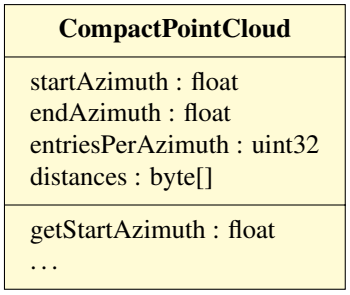
From the normal vector $\vec{n}$ and the mean point $\overline{P}$, we can again determine the Hessian normal vector. In the end, we have been able to break down the algorithm from $\mathcal{O}(m^2 n)$ to $\mathcal{O}(n)$.

6.2 Clustering

In current work with 3D-LiDAR data, the data is often first projected into a heightmap [42, 19, 25]. Then directly adjacent measurements are combined with similar measured values. Alternatively, the measurements are also summarized by means of a distance criterion. The former method has the disadvantage that individual outliers cause the object to disintegrate in several clusters. The latter is usually combined with a KD-tree or a similar data structure, which typically entails high costs for the creation ($\mathcal{O}(n \log n)$ in case of KD-Tree [5]). Since the tree has to be rebuilt after each 360 degree measurement this is an problem.

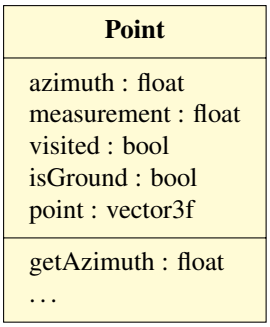
Here a method is suggested which combines the advantages of both methods. It is therefore necessary to know how the data from the OpenDAVINCI Middleware is delivered. Because the OpenDAVINCI Middleware relies on the transmission of the data with UDP Multicast, the data is transferred in a compact form that fits into a single UDP frame. The structure can be seen in fig. 6.3.

Figure 6.3: OpenDAVINCI Point Cloud Data Structure



In this case, we asume a constant rotational rate of the sensor output, which results in an equidistant distance of the measured values. Important to know is also that we always get a complete degree measurement. The number of measurements per azimuth is recorded in entriesPerAzimuth and corresponds to the Velodyne VLP-16. In order to arrive at the actual measured values, two distance values must be merged and converted to an unsigned 16-bit integer, which then contains the measurement in cm. In each case, 16 of these values result in a measuring frame in which the polar angle must be mapped to a range between -15 and +15 degrees. After the spherical data of each measurement point have been reconstructed, we rotate our coordinate system with the help of the heading of the Applanix POS LV which we add to the azimuth angle. As a result, the y-axis of our coordinate system is always aligned to the north, which greatly simplifies the subsequent tracking of the objects. Afterwards the spherical data is converted into Cartesian coordinates and stored in a point data structure, to see in fig. 6.4.

Figure 6.4: Point Data Structure



This is stored again in a static two-dimensional array: Points [2000] [16]. The order of the data is keeped thereby to allow efficient access to the values based on their azimuth angle. This data structure now provides the base for the subsequent clustering alorythm.

On this basis, a DBSCAN [15] is now executed. The DBSCAN algorithm has the following advantages. In contrast to the K-Means algorithm, for example, it is not necessary to know how many clusters exist. The algorithm can recognize clusters of any shape (e.g., not only spherical). Furthermore, DBSCAN can deal with noise and recognize as this. This makes the DBSCAN optimal candidate for us, because our objects can be have any shape and measurement from the sensor can have errors within a real scenario, which can be asumed as noise effect. In fact, DBSCAN is itself of linear complexity. However, most computing times are caused by the neighborhood calculation. But instead of the range request via a tree structure, we here profit from the behavior, that measurements in a small neighborhood have a similar azimuth angle. For this purpose, we examine two additional entries for each measured value to the left and right in our array. Therefore, we need to check $5 \cdot 16 = 80$ values. The runtime of the range request can therefore also be performed in linear complexity. All measured values previously classified as ground are skipped during the calculation. In addition, the construction of a KD-tree is omitted, which also leaves us a run-time advantage. All clusters are stored by the algorithm as a list of references to the original array to avoid unnecessary copying.

linear complexity ???????? und erklären weichen Einfluss das ganze hat!

6.3 Tracking

The task of tracking is to establish a relationship between the measured values over time. The tracking is divided into two sections. Tracking of the clusters from the DBSCAN and the creation and tracking of parameterizable objects.

6.3.1 Cluster Tracking

For cluster tracking, we assume that objects move only slowly from time step to time step, and the shape of the cluster also changes only slightly. This is important because the position of a cluster is defined by a mean value point. Tracking is performed in $\mathbb{R}^2$. In the initial step, an ascending ID is assigned to each cluster. In each further step, each new cluster is assigned the ID of the old cluster, which has the smallest distance over time. For this distance, there is a generous upper bound of 3m, which was determined experimental, clusters that are not within this boundary are given a new ID. This results in the fact that multiple clusters can be assigned the same ID, which is important because

objects sometimes break down into multiple clusters.

6.3.2 Object Tracking

The basis for the object tracking are the previously tracked clusters. In the initial step, objects are created from all clusters, with the same ID as the clusters have. Which apart from the measured values contain further parameters for the description of the object. This includes values such as the potential size and position of the object as a rectangle, the speed of the object and its direction, the ID and type of the object and its confidence. The ID is derived from the ID of the initial used cluster

In each further step, all clusters with the previously identical ID are used to update the objects. New objects are created from clusters with new IDs.

The first important step in the update process is the calculation of the movement direction of an object, because the following calculations are based on this. When calculating the direction of movement, it must be noted that the movement of our own vehicle must be separated out. The position data of the Applanix POS-LV are used for this purpose. Since both the position data of the Applanix system and the detected position of the vehicle have errors, the direction of movement is updated only with a minimum movement of 2m.

$$\begin{aligned} \Delta x &= P_x(t) - P_x(t_{-2m}) + \Delta C_x \\ \Delta y &= P_y(t) - P_y(t_{-2m}) + \Delta C_y \\ \theta &= \text{atan2}(\Delta y, \Delta x) \end{aligned}$$

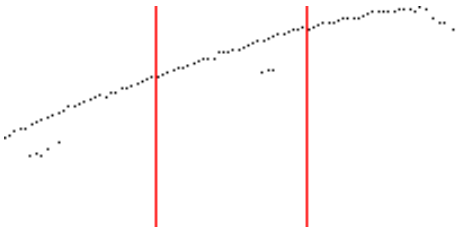
with P - position of the object, C - position of our own vehicle

The result can be viewed in fig. 6.5. It is easy to see that the direction of movement (arrow) does not conform with the orientation of the object (black), but the bounding box is correctly oriented. How this calculation comes from is clarified in the following.

Figure 6.5: Obstacle Movement

Based on the direction of movement, the alignment of the vehicle is now calculated. To do this, all clusters assigned to the object are grouped together and rotated by $-\theta$. Then the object is divided into 3 equal sized segments (fig. 6.6).

Figure 6.6: Obstacle Cutting



It is also determined whether the object is above or below the x-axis. This is important because we need to know which side of the object we are measuring. If the object is below the x-axis, then the y-value is maximized in the next step; if it is above, it is minimized. In the following, we assume that the object is under the x-axis. Therefore we maximize the y-values in the left and right segment of the divided obstacle. The division into 3 segments is necessary to prevent the two maxima from running into the same point and the distance of the points doesn't become too small, which would cause a great inaccuracy. With these points $(\vec{R};\vec{L})$ (fig. 6.7) a correction of the rotation of the object is now calculated:

$$\Delta x = R_x - L_x$$
$$\Delta y = R_y - L_y$$
$$\theta_{correction} = \text{atan2}(\Delta y, \Delta x)$$

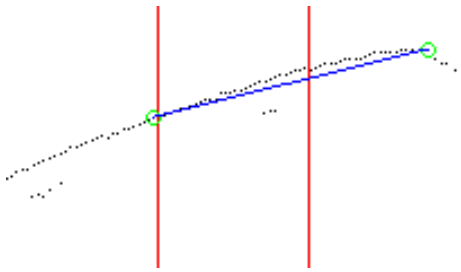


Figure 6.7: θ - Correction

warum 0.5m??

After applying the correction, the size of the obstacle is calculated. For this purpose, the maximum and minimum x and y values of rotated objects are used. With these values, a histogram for the length and width of the obstacle, rounded to 0.5 m, is now established over time. The most probable value is then selected. This causes the size of the obstacle to change more often before the size converges to a stable value. Since the size of the object the begin can be very small, there is a lower limit of 0.9m for both values. Measured values for a sample object can be seen in fig. 6.8.

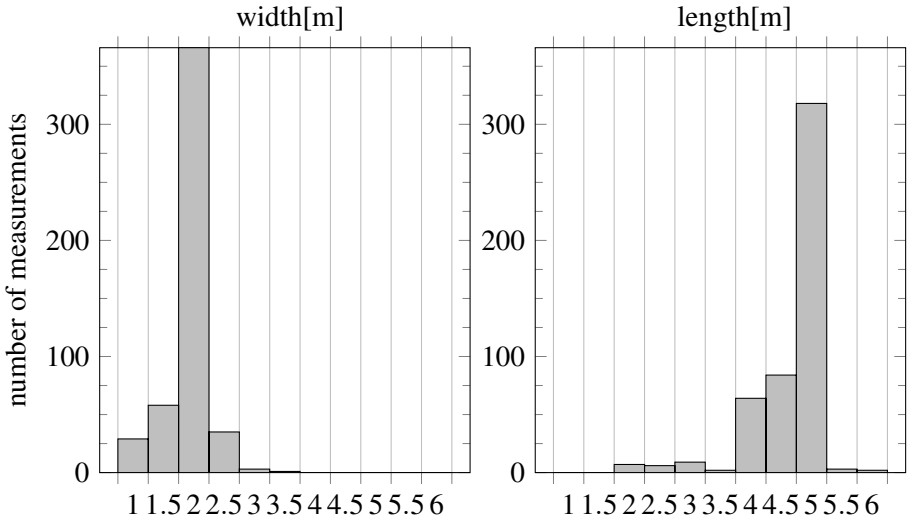


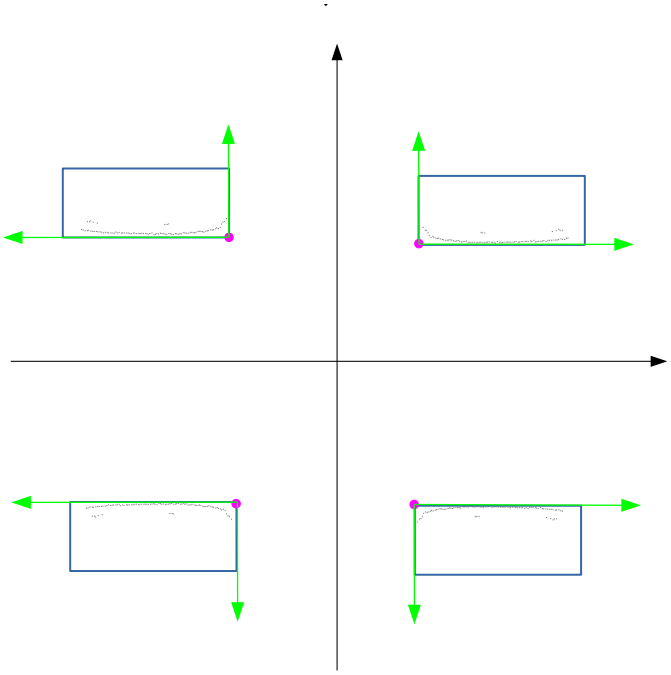
Figure 6.8: Object Size Histogram

Easy to see, a width of 2m and a length of 5m is calculated for the object. In

this case the object is a Volvo S60, which has outer dimensions of approximately 1.9m and 4.6m, whereby the deviations are correctly within, the rounding of the values. It can be seen that the distribution of the width, is similar to a normal distribution, which is due to the fact that the rotation of the object has a noise. In the case of length, however, we see that the algorythm tends to calculate to small values, which results from the fact that this can only be inadequately measured when the vehicle is moving towards the sensor. Because to the internal vehicle structure, the sensor is not able to see through the vehicle, so the values are to small.

The Bounding Box of the object is now calculated from these values. To this end, a further distinction must be made. If the object is to the left of the y-axis, the maximum x and y value is used as an anchor point and the object is stretched to the lower left using the calculated length and width. However, if the object is to the right of the y-axis, the minimum y-value is used and the object is stretched to the lower right. What the whole graphically means is illustrated in fig. 6.9 for all four cases .

Figure 6.9: Bounding Box Calculation Cases

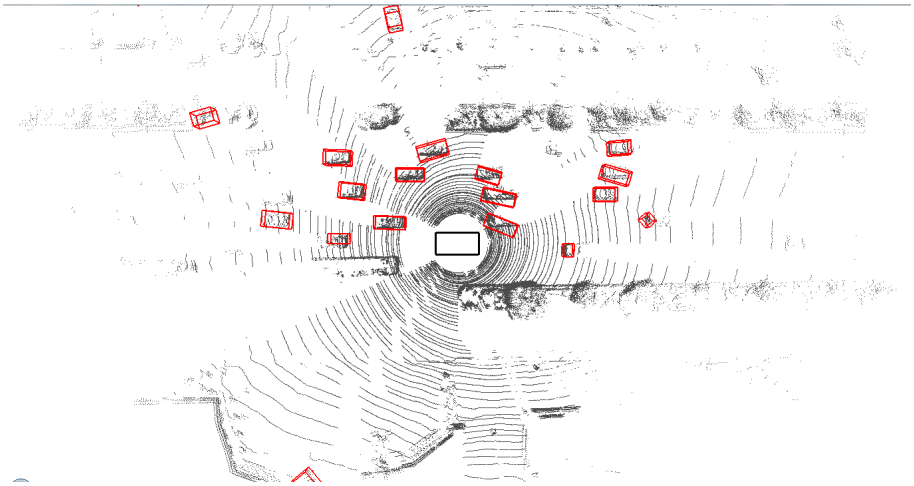


For the subsequent filtering of the measured values with the help of an extended Kalman filter, the position of the vehicle from the center of the bounding box is now determined, using the mean value of the four corner points.

However, the position used for the calculation of the direction of movement is different, since the so calculated position is not yet available at this time and the position is very unstable shortly after the initial recognition due to the frequent dimensional changes. Therefore, the maximum x-coordinate of the rotated clusters is always used as the position. This position can also be examined in fig. 6.5, as a small green circle. Since θ in the initial time step is zero, this corresponds to the global maximum x-coordinate of the cluster. This leads to the assumption that the object moves in the positive x direction in the initial time step. For objects where this is not the case, this leads to a short-term oscillation of the orientation, which however quickly stabilizes over time.

One thing or another may be wondering why the bounding box was calculated so elaborately. A simple way to calculate a bounding box for the objects would be the calculation of the minimum bounding box over the convex hull, as it is done in many other works [42, 19]. The minimum bounding box, however, under certain circumstances does not provide the desired result. On the one hand you have only the size of the current measurement and on the other hand it can provide a wrong orientation as seen in fig. 6.10.

Figure 6.10: Error with minimum Boundingbox [42]



However, since the orientation and position of the objects are used as input for the subsequent Kalman filter, which reacts very sensitively to false orientations, the algorithm was developed.

6.3.3 Object Confidence

In order to prevent the detection of false positives immediately recognized as an object and thus to influence the subsequent logic, a confidence value is now introduced. Before a recognized object is considered to be valid, this must achieve a certain confidence value. The initial confidence value of an object is one. The confidence value is increased by one if the object can be tracked in two consecutive time steps and satisfies the following conditions:

- The width of the object must be less than the length of the obstacle plus 1.5m
- The length of the obstacle must be less than 10m
- The width of the obstacle must be less than 4m

If one of these constraints is not met, the Confidence value is halved. For an object to be considered kind, it must reach a confidence value of three so that an object can only be recognized after at least two iterations. This value was empirically determined and represents a tradeoff between fast detection and filtering.

6.4 Classification

implementierung Plausibilitätstest

Now a classification of the objects is made by their size, there is no classification according to moved and immobile objects. Only pedestrians, cyclists,

vehicles, and others are differentiated. The size of the objects is used as a classification criterion. The classification is as follows:

pedestrian: length < 1.5 and width < 1.5

cyclist: length < 2 and width < 1.5

car: length < 10 and width < 4

undefined: length >= 10 and width >= 4

Furthermore, a plausibility test is carried out using the speed. Thus, a pedestrian must not exceed a speed of 10km/h and the maximum speed for a cyclist is 30km/h. Since no sensible speed limit can be assumed for vehicles, we use the change of orientation in their place. The maximum rotation rate is assumed to be 0.1 rads/sec from [21]. Since the value is an upper limit, we assume a slightly higher value of 0.2 rads/sec. If one of these values is exceeded, the confidence is also halved.

6.5 State Estimation

Ergänzen Kalman Filter: Normalveteilung müsste überprüft werden, aber da der EKF für Tracking state of the art ist.. nehmen wir an, das dass funkiniert. referenzen dazu suchen !!!

Following the tracking, a state estimation is performed on the detected objects. This is necessary because objects can be hidden during the movement, be it by other moving objects or buildings. An estimation of the state beyond the detection horizon allows us to make a statement about the position of objects which are not visible at the moment. Furthermore, it allows you to easily redetect objects that have not been detected at a time, ie to assign the same ID to the object as before and allows a well filtering of the measured objects.

From the previous tracking, we can get the current position, speed, rotation and rotation speed. For an estimation of the state with the for vehicle common the bicycle model [1, 34, 22], we miss the wheelbase and the weight of the vehicle. Therefore, we must limit ourselves to a relatively simple “Constant Turn Rate and Velocity” model. This allows us to use the same model for all classes of objects. Because this model can also be used on pedestrians and cyclists.

6.5.1 Constant Turn Rate and Velocity Model

The state vector [33] of the CTRV model looks as follows:

$$\vec{x}(t) = \begin{bmatrix} x & y & \theta & v & \omega \end{bmatrix}^T$$

x - x-Axis

y - y-Axis

θ - Object Yaw Angle

v - Object Velocity

ω - Yaw Rate

The dynamic matrix is obtained by a non-linear state transition:

$$f = \vec{x}(t+\Delta t) = \begin{bmatrix} \frac{v}{\omega}(-\sin(\theta)+\sin(\Delta t\omega+\theta))+x(t) \\ \frac{v}{\omega}(\cos(\theta)-cos(\Delta t\omega+\theta))+y(t) \\ \omega\Delta t+\theta \\ v \\ \omega \end{bmatrix}$$

Since the dynamics matrix is a non-linear state transition, we still need the Jacobian matrix. This was shortened because of its size.

$$J(\vec{x}) = \begin{bmatrix} 1 & 0 & \frac{v}{\omega}(-\cos(\theta)+\cos(\Delta t\omega+\theta)) & \dots & \dots \\ 0 & 1 & \frac{v}{\omega}(-\sin(\theta)+\sin(\Delta t\omega+\theta)) & \dots & \dots \\ 0 & 0 & 1 & 0 & \Delta t \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

6.5.2 Extendet Kalman Filter

The Extendet Kalman Filter (EKF) [32] is the nonlinear version of the Kalman filter, which is linearized by an estimate of the current mean and the covariance. In addition to our model, further values are required. For this purpose, we look at the calculation rules of the necessary steps.

Prediction

In the first step of the filtering process, the preceding estimation of the state dynamics is projected to a prediction for the current time.

$$\hat{x}_k = f(\hat{x}_{k-1}, u_{k-1})$$
$$P_k = F_{k-1}P_{k-1}F_{k-1}^T + Q_{k-1}$$

Where $\hat{x}_k$ is the estimate of our state for the current time step and P_k is, because of the prediction, our increased covariance.

Since we can not know the control variables of the other objects, we asume u to be zero for our calculations. For the state transition model F we use our previous calculated Jacobian matrix applied to the previous state.

$$\hat{x_k} = f(\hat{x}-k-1, 0)$$
$$P_k = J_{k-1}P_{k-1}J_{k-1}^T + Q_{k-1}$$

For the Process Noise Covariance Matrix Q, we assume typical worst case values for vehicles from [21] as the starting point, maximum acceleration: $a = 8.8\frac{m}{s^2}$, maximum turn rate: $\omega = 0.1\frac{rad}{s}$, maximum turn rate acceleration: $\dot{\omega} = 0.1\frac{rad}{s^2}$. These values are then integrated until the unit is correct. Finally, since we want to have a variance, the values are squared. We assume no correlation between all other values and get only values in the main diagonal.

$$Q = \begin{bmatrix} (\frac{a\cdot\Delta t^2}{2})^2 & 0 & 0 & 0 & 0 \\ 0 & (\frac{a\cdot\Delta t^2}{2})^2 & 0 & 0 & 0 \\ 0 & 0 & (\omega\cdot\Delta t)^2 & 0 & 0 \\ 0 & 0 & 0 & (a\cdot\Delta t)^2 & 0 \\ 0 & 0 & 0 & 0 & (\dot{\omega}\cdot\delta t)^2 \end{bmatrix}$$

Furthermore, some optimization had to be made in this step in order to ensure the numerical stability. If the yaw rate is very low, we risk a division by zero. In order to except this situation, a new state transition was defined, in which we ignore the direct influence of the yaw rate on the position, when it falls below a value of $0.0001\frac{rad}{s}$. The resulting error is very small, since the yaw rate is close to zero.

$$f = \vec{x}(t+\Delta t) = \begin{bmatrix} v\cdot\Delta t\cdot\cos\theta + x(t) \\ v\cdot\Delta t\cdot\sin\theta + y(t) \\ \theta + \Delta t\cdot\omega \\ v \\ \omega \end{bmatrix}$$

The corresponding Jacobian matrix is then:

$$J(\vec{x}) = \begin{bmatrix} 1 & 0 & -\Delta t\cdot v\sin\theta & \Delta t\cdot\cos\theta & 0 \\ 0 & 1 & \Delta t\cdot v\cos\theta & \Delta t\cdot\sin\theta & 0 \\ 0 & 0 & 1 & 0 & \Delta t \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Updating

correction step EKF näher beschreiben da der nicnt standard im ekf is was macnst du mit statevector (wohl nur werte überschreiben) was machst du mit uncertainty so den state estimation teil hab ich durchgelesen

The predictions are finally corrected with the new information of the measurements.

$$\begin{aligned} \tilde{y}_k &= z_k - h(\hat{x}_k) \\ S_k &= HP_kH_k^T + R_k \\ K_k &= P_kH_k^TS_k^{-1} \\ \hat{x}_k &= \hat{x}_k + K_k\tilde{y}_k \\ P_k &= (I - K_kH_k)P_k \end{aligned}$$

Since we directly measure all states of our system there is observation function for our measurements, means our $h(\hat{x}_k)$ equals $\hat{x}_k$. Because of this our observation matrix H results in a simple Identity matrix.

For the Measurement Noise Covariance Matrix R we assume experimental determined values. Since we are quite sure about the position we assume a small value of 1 and a value of 0.1 for the rotation, for the same reason. For the velocity we chose a quite high value of 4, because it is derived from the position over time. The same for the yaw rate with a value of 0.1. Again we assume no

correlation between all other values, which it is most likely not true, but would need more further investigation.

$$R = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0.1 & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 0 & 0.1 \end{bmatrix}$$

In this step of the filter it must be noted that we are working with θ that is defined in the range of $[-180^\circ \leq \theta \leq 180^\circ]$ and therefore singularities can occur. Because our measurement data is noisy, our state may be at 179 degrees and is updated with a value of -179°. For the sake of simplicity we assuming that both values are equally weighted, the update step results in a value of 0 degrees, which is obviously wrong. Therefore, before each updating step, 360 degree is added or suptracted, so the distance between state and measured value is minimized. After updating, the internal state is again normalized to the range $[-180^\circ \leq \theta \leq 180^\circ]$. Consider now again our example with the corrected values, state: $179 - 360 = -181$ degree and measured value: -179 degree, we get the correct value of -180 degree.

Initial Uncertainties

Our initial covariance matrix P was also determined expirimentel. For the position we take a high degree of safety, since we can measure them directly, and put them on one. Since we also measure the rotation directly we take a value of 0.1. Since we can not measure the time-dependent variables in the first step, a very high value of 1000 is chosen here.

$$P_{initial} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0.1 & 0 & 0 \\ 0 & 0 & 0 & 1000 & 0 \\ 0 & 0 & 0 & 0 & 1000 \end{bmatrix}$$

6.5.3 Reassigning

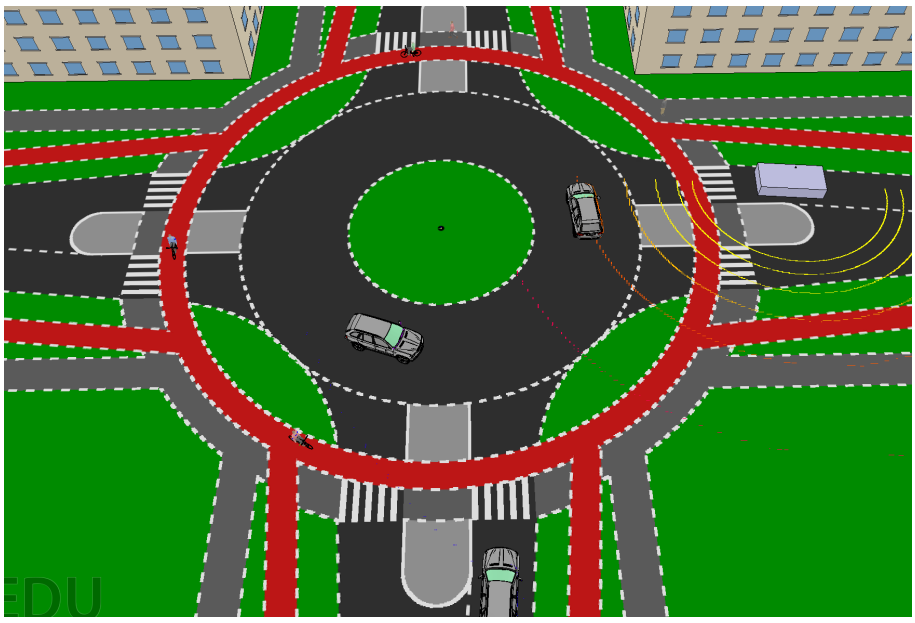
If an object is within the range of the Velodyne sensor and is not detected in the subsequent time step, the prediction step of the Kalman filter is still executed. This is done as long as the confidence value of the object does not reach zero. As soon as the cluster tracking detects a new object which can not be assigned a previously known ID, the position is matched with all objects in the prediction phase. If the new object is within three meters to the predicted position, the cluster is assigned to the object and the correction step of the EKF is performed. If more than one object matches these condition the closest object is assigned.

7 Simulation

The simulation environment being used is VREP [1]. VREP has been developed for various robotic applications. VREP allows you to construct any multi-body simulation (MKS) based on various physics engines within a graphic editor. Furthermore, VREP already has many ready-to-use sensor models, such as the Velodyne VLP-16. The whole simulation can communicate via a RemoteAPI interface with nearly every programming language.

7.1 Simulation Scenario

Figure 7.1: VREP



referenzen zu "roundabout in law" mit formelsymbolen und so

For the simulation scenario, a simple small roundabout with an outside diameter of 30m, according to the rules in [36] was designed. According this the cycle path must have a minimum distance of 4 m to the road and the lane width

1. http://www.coppeliarobotics.com/

of the circular driveway and circular exit should be 3.75m. The recommended corner rounding radius is according to [36] 12m.

The small roundabout was chosen because this is the most interesting object due to its size in combination with the VLP-16, which still has an acceptable resolution in this area. And in urban areas due to their size, they are frequent. Furthermore, due to its size, it is possible to survey the entire roundabout. The whole scenario was down-scaled by a factor of 10 due to limitations in Vrep.

Inside the scenario there are three bicycles, one pedestrian and three other vehicles driving around the roundabout. The objects are moving on fixed paths. The speed of all traffic is adapted to the type. The pedestrian moves with the 5km/h usually asumed for pedestrians. The bike with 15km/h. The two cars move at different speeds between 25 and 35 km/h. In order to avoid a collision of the vehicles, both vehicles are equipped with distance sensors at the front. If this sensor detects a vehicle, the vehicle enters an Adaptive Cruise Control mode and adapts to the front vehicle. The Adaptive Cruise Control is implemented as a simple proportional controller.

The autonomous vehicle also moves along a fixed path, which leads from the right into the roundabout and leaves the roundabout at the third exit. The task of the vehicle is to safely enter the roundabout and safely leave the roundabout. In doing so, the vehicle must pay attention both to the pedestrians, the cyclists and the other vehicles within the roundabout. For this, the vehicle is equipped with a virtual Velodyne VLP-16 and the previously described algorithms deliver the detected objects.

7.2 Simulation Logic

In order to carry out the previously described scenario, a logic had to be developed, which includes not only the state machine for passing through the roundabout, but also the localization of the vehicle in the scenario as well as the connection of all sensors to the object detection. Since all subsequent calculations require little resources, the corresponding software was developed in Python.

7.2.1 Sensor Connection

The object detection requires the data of the Velodyne VLP-16 and the data of the Applanix POS-LV as sensor input. However, only the current position in the WGS84 format and the current heading are required by the object detection. Since the Applanix sensor can not be easily reconstructed in VREP, the necessary data is simply generated from the position and rotation data readable in Vrep. Since the position in VREP is specified in Cartesian coordinates, they are transformed via the transformation contained in OpenDAVINCI, using a reference coordinate in WGS84 format.

The connection of the Velodyne VLP-16 is more difficult because the structure of the measured data differs significantly from the original Velodyne. Although the data is also output in polar coordinates, only measured values are output when they hit an object. Since a large part of the scenario is empty space, the measurement data have many holes on them. But the object detection needs continuous measurements, so they must be transformed into a suitable form.

It is important to note that object detection requires a complete 360 degree

measurement. To do this, you must wait until all necessary measurement data are available. The virtualle Velodynle runs a measuring rate of 10Hz, and is divided into 4 segments, which are read one after the other. The time step used for the simulation is 50 ms, so that we can read out 2 segments in one step. Once all data are collected, they are assembled into a suitable measuring frame.

The measured data are provided as a list of spherical coordinates (radius r, polar angle θ, azimuth angle Φ). Since the azimuth and polar angles are not at equidistant intervals but have a much higher resolution than the original sensor, the measured values are rounded down to the original resolution of 0.2 and 2 degrees and the measured values are entered into a corresponding two-dimensional fixed-size array. Thus, areas that are not detected have the value zero.

After all necessary measurement data have been collected and converted, they are sent to the object detection via a specially developed OpenDAVINCI-Python interface. Unnecessary data for the Applanix is simply set to zero.

7.2.2 Mapping

The OpenDAVINCI internal Compressed Scenario Data Format (SCNX) [6] is used for mapping. This allows scenarios to be defined by describing and combining stationary and dynamic elements to formulate different traffic situations. The SCNX format offers, among other things, the classes ROAD, LANE for the modeling of roads. One ROAD can consist of several LANEs. A lane consists of a set of points which describe the course of the lane. The lane can be assigned with the attributes lane mark and width. Individual lanes can be connected to each other. What makes it possible to build a complicated road network with these simple models. Since OpenDAVINCI does not provide a Python interface for parsing the Scenariofiles, a special Lexer was implemented for the processing of the Scenariofiles. At thise time, only the classes required for the simulation are implemented. Therefore, a lane can only be described by a point model, while OpenDAVINCI offers additional models.

Furthermore, the model included in OpenDAVINCI had to be extended, since OpenDAVINCI unfortunately does not support other types of paths. Therefore, the class lane has been extended by one type attribute, which makes it possible to declare this as a bicycle or footpath. The structure of the extended Python classes can be examined in fig. 7.2.

For the handling of the roundabout, however, the map format had to be further extended. According to RASt [38], a roundabout should be as circular as possible, for the sake of simplicity, we assume that the roundabout in simulation is a perfect circle. This means that both roads, cycle paths and footpaths are perfectly circular and can be described by an inner and outer radius. For this reason, a class ROUNDABOUT [fig. 7.3] was added to the description of the roundabout, which contains the center point of the roundabout, references to all lanes, and inner and outer radii. Furthermore, the connections to the connecting lanes are defined therein.

Localization

As we are within simulation, we can assume a perfect position for localization. Localization within the map means that we want to map the current known po-

Figure 7.2: Parser Objects

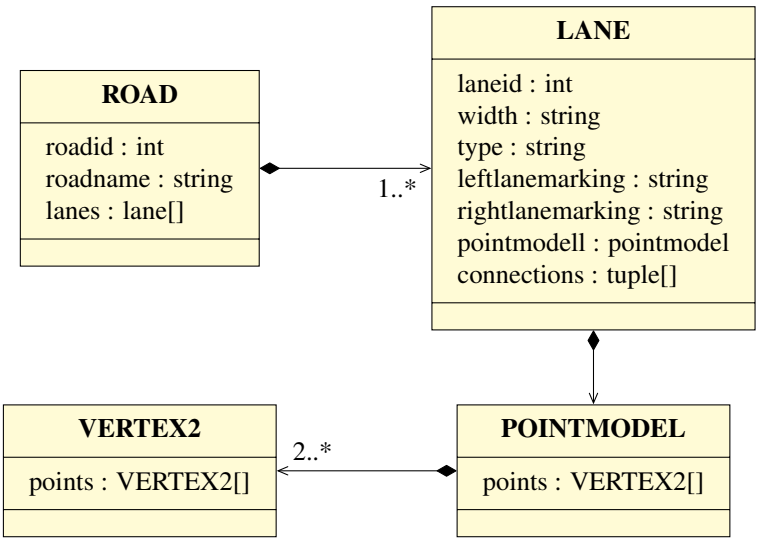
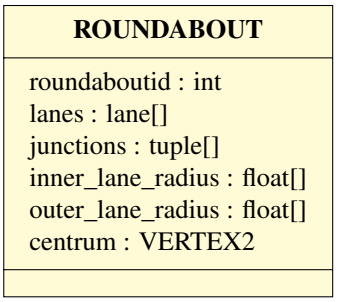
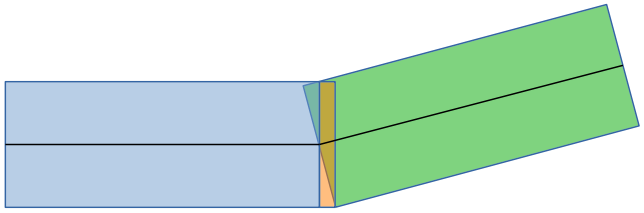


Figure 7.3: Roundabout Class



sition to a lane segment. Since the lanes are only given in the form of points and width, rectangles are calculated for each lane segment and the current position is checked against each rectangle. This is done by rotating the rectangle (and the position) to an axis-aligned position and moving it to the coordinate origin. Furthermore, the rectangle must be enlarged if we do not process the last line segment, since otherwise a gap would remain between the rectangles. For this purpose, the angle to the next line segment is used. The whole situation is symbolized in fig. 7.4. The blue rectangle represents the current one to be edited, the green rectangle the next and the orange the necessary extension.

Figure 7.4: Mapping Rectangles



As seen in fig. 7.4, the individual segments are usually not at a 180 degree angle to each other. Therefore, the rectangles can overlap and more than one rectangle can be returned for one position. In this case, the direction of movement from the object is used and the most recent segment of the direction of movement is selected. If several lanes cross over, for example, cycle paths that intersect the roundabout, the correct segment is selected based on the object type, ie: auto -> road or cyclist -> cycle path. Furthermore, the x-coordinate

of the object in the rotated and shifted coordinate system is returned, which is necessary for the distance calculations. We always assume that the object is in the middle of the track. Alternatively, the “ Ray Casting Algorithm ” [17] can be used for this problem. Since we assume for all distance calculations that the object is in the middle of the track, this calculation results in a very small error and the distance is practically derived from the previous calculations so that it was not applied.

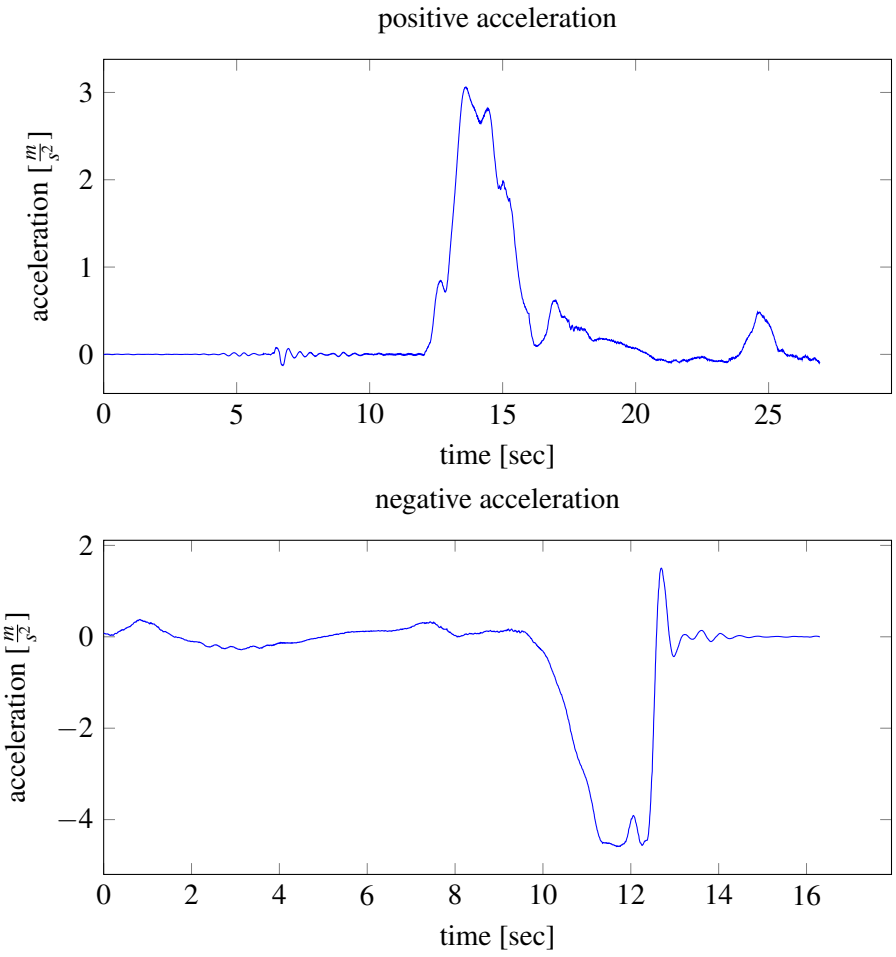
7.2.3 State Machine

nummer anpassen

The state machine is implemented according to Gang of Four [24]. For this, X States were implemented. A constant acceleration model is assumed for all vehicle computations. In order to determine the maximum positive and negative acceleration, test runs with the real vehicle were recorded and the maximum subjectively pleasant values being determined.

The used recordings can be seen in fig. 7.5, the vehicle was accelerated from 0 to 25 km/h or braked from 25 to 0 km/h. The maximum values were read, so we assume $3m/s^2$ for the maximum positive acceleration and $-4.5m/s^2$ for the maximum negative acceleration. The maximum speed of the simulated test vehicle is set to 30 km/h.

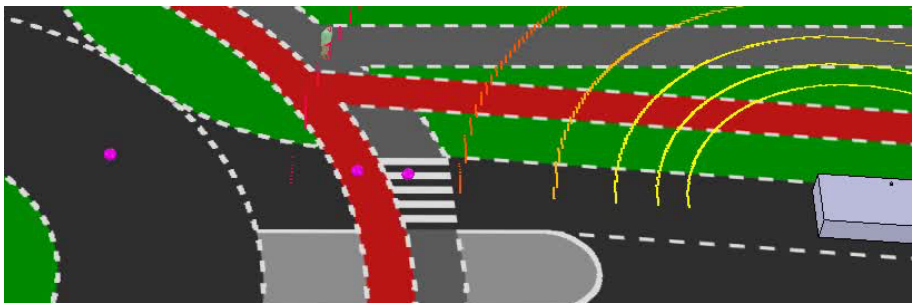
Figure 7.5: Acceleration Plott



Before each call of a state, all objects delivered by the object detection are picked up and mapped to the lanes held in the map. Objects that were not on a track or path are rejected. Furthermore, the expected intersection points of our

own vehicle with all paths in the roundabout is calculated (see magenta points in fig. 7.6). This is necessary because the vehicle is not driving straight into

Figure 7.6: Intersection Points



the roundabout and the position in the map can be slightly corrupted. For this purpose, the intersection points of trajectory and lanes are calculated from the expected vehicle trajectory and the simplified roundabout model.

Start - State The vehicle is in the Start - State until it reaches a distance of 20 m to the outside lane of the roundabout, afterwards the state is changed to the ToRoundabout - State. In this state the vehicle is driving at a maximum speed of 30 km/h

ToRoundabout - State We are in this state, if the vehicle is not yet located in the roundabout. At each call, the remaining distance, of all obstacles wich are moving in the direction of the intersection, to the intersection point is calculated. Based on this distances and the object speeds the estimated time window is calculated, in which the obstacles reaches the intersection point. A time window is also calculated from the maximum acceleration and speed of our vehicle and the calculated distance to the intersection point. If this time window is smaller than one of the obstacles plus a safety interval of two seconds, the vehicle enters the roundabout and changes to the InRoundabout - State. If this condition does not apply, the vehicle enters the Brake - State.

speed durch velocity ersetzten

Brake - State In this state, the distance of our vehicle to the next intersection point is calculated. With this distance, the necessary negative acceleration is now calculated in order to get the vehicle standing up to this point, and sent to the vehicle. After the vehicle has been brought to a standstill, we changes back into the last called state.

InRoundabout - State When the vehicle is inside the roundabout, it carries out Adaptive Cruise Control (ACC). That is, the vehicle travels at a constant speed until it encounters an obstacle. As soon as an obstacle is reached, the speed is reduced proportionally to the distance of the obstacle, so that a safety distance is established. If the vehicle feeds the target exit at 10m, the state will be changed to ExitRoundabout. The position from the target exit is defined in the map.

radweg und fußweg vereinheitlichen

ExitRoundabout - State The ExitRaoundabout - State checks whether there are obstacles on the wheel or footpath within a given range. If this is the case,

the vehicle changes to the Brake - State and stops in front of the cycle track. The vehicle does not move until the specified range is free again. The vehicle remains in the ExitRoundabout - State until the vehicle is reset to the starting position by the simulation. The vehicle then changes to Start - State.

8 Evaluation

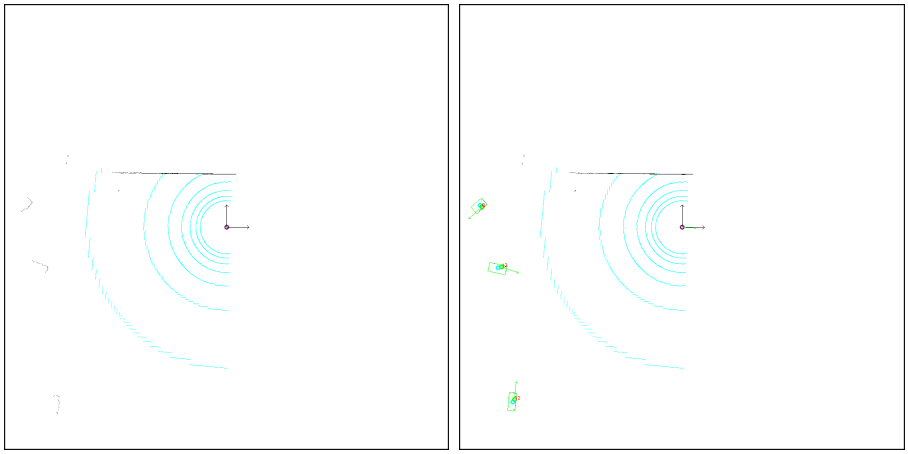
We start with the evaluation inside of the simulation. This allows us to make a statement about the calculated values such as position and speed, since the real values can be easily read within the simulation.

8.1 Simulation

8.1.1 Detection Distance Performance

First, let's look at the general perception performance of the algorithm, we compare the first two steps of the simulation. These are shown in fig. 8.1. The figures represent a range of 120x120 meters, so that we cover slightly more than half the measuring range of the sensor. In the first step, nothing is recognized since the algorithm requires at least one step in time to raise the confidence value of the detected objects over the detection threshold.

Figure 8.1: Detection Distance Performance



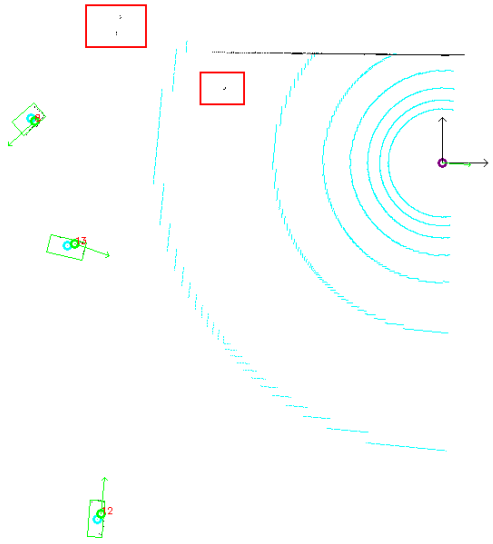
Timestep 1 Timestep 2

In the second step we see the three objects recognized. All of these objects are vehicles. The vehicle in the lower left side has a distance of 68 meters to the test vehicle and was correctly recognized and classified. The color of the boxes represents the type of classifying, green boxes are vehicles, blue boxes cyclists.

The maximum distance at which the vehicle can be fully detected can be determined within the scope of the simulation with close to 95m. Which is limited by the range of the sensor. Whether the vehicle is classified correctly depends on the orientation of the vehicle. If the vehicle moves straight to the test vehicle, only the front of the vehicle can be seen. This means that the vehicle has an incorrect size and is recognized as a cyclist or pedestrian. Since a histogram is used for the calculation of the size, this false classification is existing until the histogram maximum converges to the correct size.

Let's take a closer look at the second time step. In fig. 8.2, we can recognize that both pedestrians and cyclists, although visible in the data (red boxes), are not recognized. This is because they do not exceed the *minPts* threshold of the DBSCAN. This is only the case when they are closer to the test vehicle, more precisely within the 6th sensor measuring sensor (about 23m).

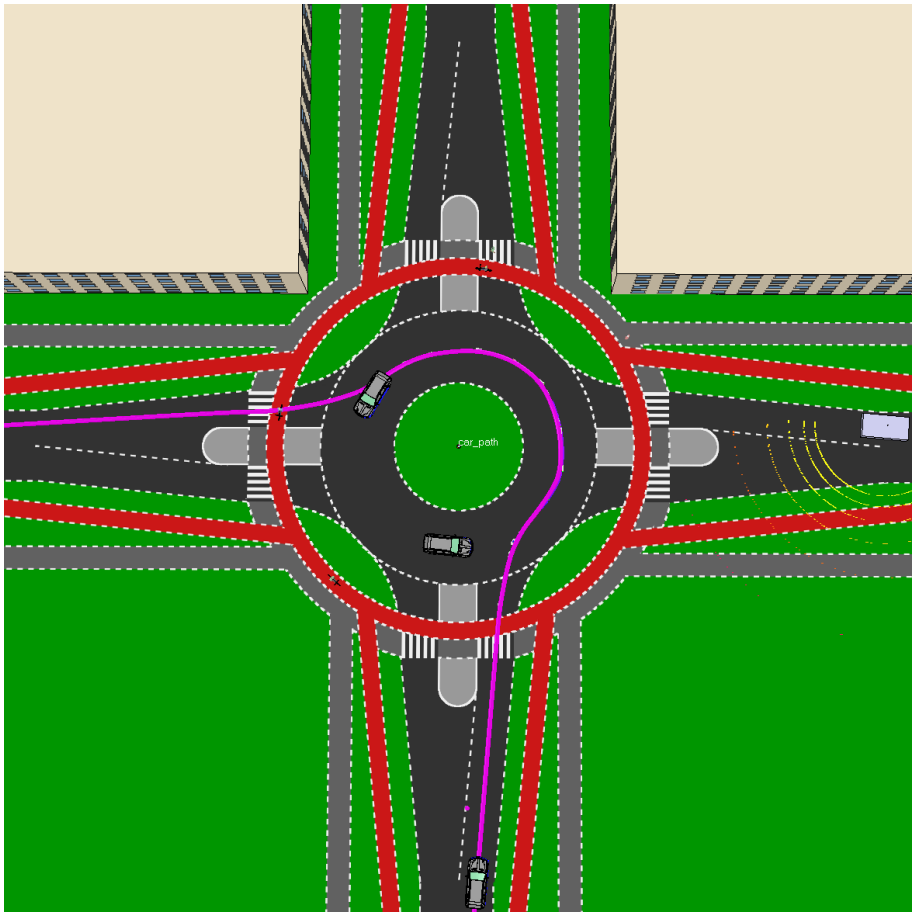
Figure 8.2: Detection Distance Performance Pedestians/Cyclists



8.1.2 Measurements Performance

In this section we evaluate the quality of the measurable variables, like speed and position of the objects in the simulation. For this purpose a vehicle which was constantly visible during the simulation is chosen. This is the only vehicle, which is not always within the roundabout, but enters and leaves the roundabout again. The path of the car is shown in fig. 8.3. The position error was calculated from the Euclidean distance of the position within the simulation and the position detected.

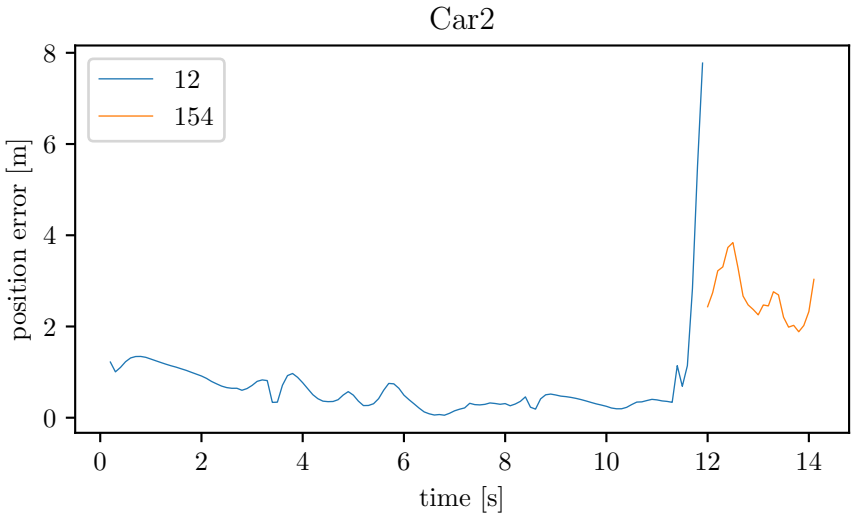
Figure 8.3: Car Path



Position Error

Let's look at the position error over the time in fig. 8.4. The legend of the figure shows the respective ID of the tracking. It can be seen that to the vehicle, two different ID's have been assigned, which means that the tracking has once lost the vehicle.

Figure 8.4: Car Position Error



Looking at the data more precisely, we see that the position, in the range where

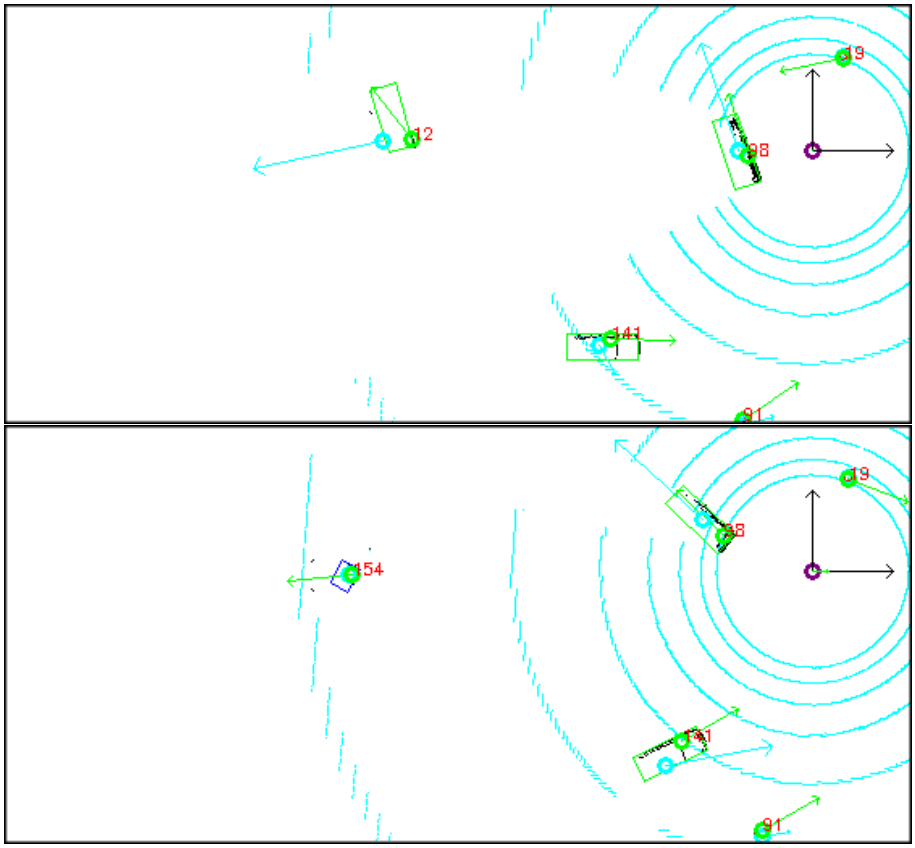
the vehicle can be detected stably, has an error of about 0.8 meters. In the moment, the vehicle can no longer be tracked, the error increases rapidly until tracking detects the object as unfavorable. Furthermore, it can be seen that after the vehicle has been assigned a new ID, the position error is generally higher.

The reason of the large error in the position lies, among other things, in the behavior of the Kalman filter. As seen in fig. 8.5, the position of the filter (light blue circle) in a vehicle within the roundabout is always slightly outside the center of the rectangle. This is because the filter works with a Constant Turn Rate and Velocity (CTRV) model, which does not quite reflect the behavior of the vehicles. Because these are not in the direction of their current orientation, but rather in their driving dynamics.

Figure 8.5: Kalman Error

Let's look at fig. 8.6 in the following section, why the vehicle went lost and the position error has risen.

Figure 8.6: Tracking Error

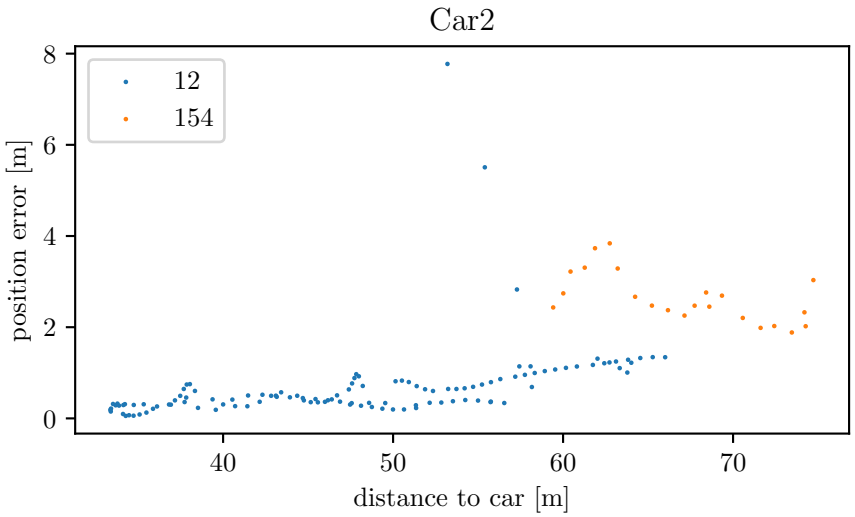


It can be seen that the object is hidden by the object 98, which means that both the position and the orientation of the object are faulty. The Kalman filter (light blue circle and arrow), however, roughly maintain the direction of movement. However, the confidence value of the object is not sufficient to trace the object back to the point in time until it is visible again. In the second part of the

figure, the vehicle will be rerecognized. Since we now can only see the vehicle from behind, the size of the vehicle is detected wrongly and it is classified as a bicycle. Since the position of the vehicle is calculated from the center of the rectangle, this leads to a systematic error. We also see that we get measurement in the front area of the vehicle. These are the wheels of the vehicle, which we can see due to the steep angle of measurement. However, since the distance to the rest of the object is too large, they are not added to the object.

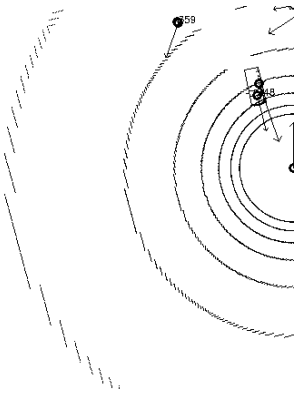
Since the pure view of the position error is not very meaningful, we now look at the error depending on the distance to test vehicle in fig. 8.7.

Figure 8.7: Car Position Error / Distance



If we ignore the position error of object 154, we see that the error increases constantly with distance. What appears at first look intuitively logical, however, if we remember fig. 8.2, we see, the object will be recognized correctly, even in the case of greater distances. The constant rise of the error is to be found in a limitation of the simulation software.

Figure 8.8: Simulation Sensor Resolution

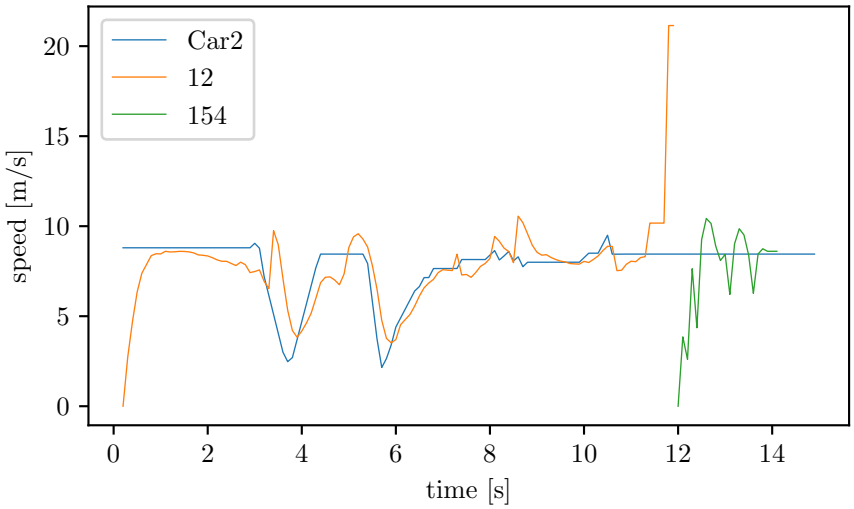


On fig. 8.8, the resolution of the sensor readings decreases with increasing distance, which leads to a rastering of the measuring circuits. This is due to the fact that it is simulated with the aid of OpenGL, which works with fixed point arithmetic. In the last measurement layer of the sensor, visible in the image , the resolution of the sensor is only 2m. This property obviously does not occur with the real sensor. Unfortunately, no suitable measuring data were recorded for the real Velodyne sensor, which would enable an evaluation to be carried out.

Velocity Error

Since, apart from the position, the velocity of the objects is particularly important for the simulation scenario, we compare the velocity of the objects detected with the real values in the simulation. For this purpose, the measured values are displayed graphically in fig. 8.9. The graph Car2 represents the ideal values from the simulation.

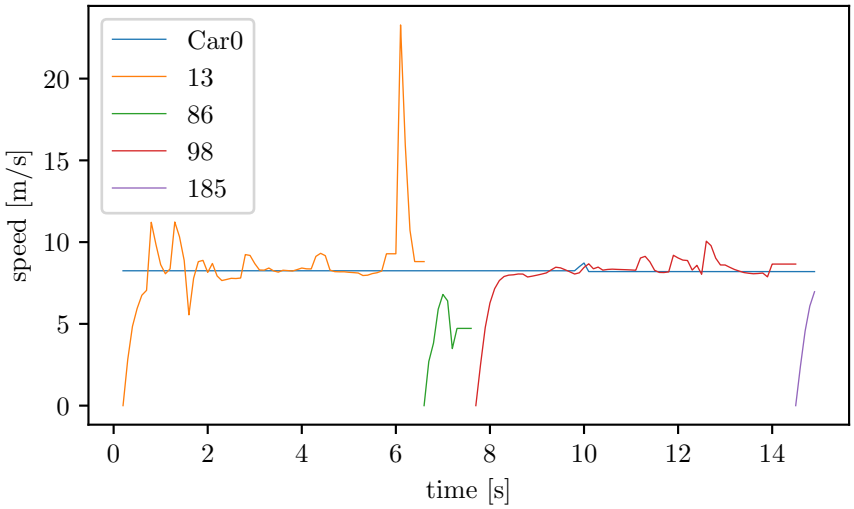
Figure 8.9: Car2 velocity



Since the velocity of the objects is derived from the position, it is zero in the inital time step, which is clearly shown in the figure. Furthermore, it is also found that the measured values follow the ideal values, but are delayed. This can also be attributed to the derivative of the velocity from the change in position. However, it is also based on the CTRV model used by Kalman filter, which assumes a constant speed. For this reason, the Kalman filter acts here as a simple lowpass and delays the values additionally. Outliers, however, also occur here when tracking is in the concept of losing the object.

Consider another vehicle, with a constant speed in fig. 8.10.

Figure 8.10: Car0 velocity



Here it is easy to see that the values are always close to the real value after a short settling time. The larger differences during the first three seconds are due to an unstable detection of the vehicle due to an unfavorable angle of the vehicle to the test vehicle. However, an optimum recognition of the vehicle is always given, particularly in the period of seconds 8 to 14, and the here measured values have only a slight deviation.

Classification

Für die Berwertung der Klassifizierung wurde ein Simulationsdurchlauf mit einer länge von 3min und 11 Sekunden aufgenommen (1917 samples), und automatisch ausgwertet. Aus diesen Daten wurde fig. 8.11 erstellt.

Figure 8.11: Classification

Object Name	Car [%]	Bike [%]	Pedestian [%]	Unclassified [%]
Pedestian	0	0	100	0
Car0	100	0	0	0
Car1	100	0	0	0
Car2	98.2	0.73	0.98	0
Bike0	0	0.5	99.5	0
Bike1	0	0	100	0
Bike2	1.9	1.2	96.9	0

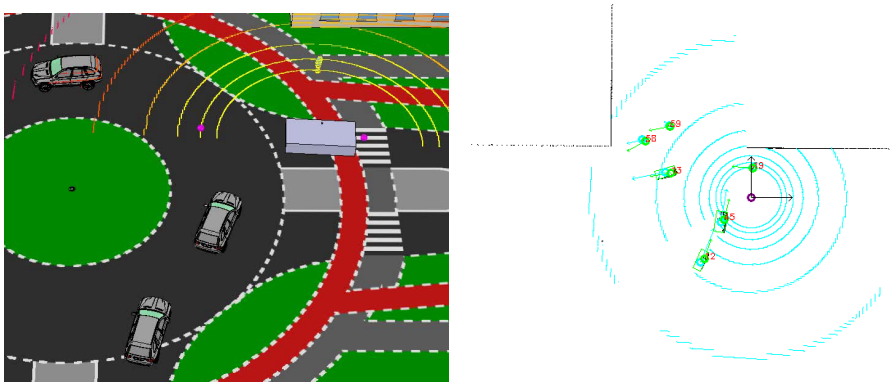
In fig. 8.11 einfach zu erkennen ist, das die Autos als auch der Fußgänger mit hoher zuverlässigkeit klassifiziert werden. Lediglich Car2 wird wenige male flasch klassifiziert. Was bereits in bezug auf fig. 8.6 gekärt wurde. Leider ist auch zu erkennen, das die Radfahrer nahzu immer flasch klassifiziert werden. Dies liegt daran, das die Fahrräder nicht genug fläche für den Sensor bieten, so dass der Großteil der Messungen nur auf der Person auf dem Fahrrad liegen.

Scenario Handling

Im folgen werten wir das Simulationsscenario an sich aus, idem wir die beiden krittischen Situationen das einfahren und ausfahren des testfahrzeuges aus dem Kreisverkehr innerhalb der Objektdetection Ansehen.

Betrachten wir als ersten die einfahr situation. Fahrzeug Auto hat bereits am Fahrbahnrand angehalten. In fig. 8.12 können wir sehen, dass alle Fahrzeuge korrekt erkannt wurden.

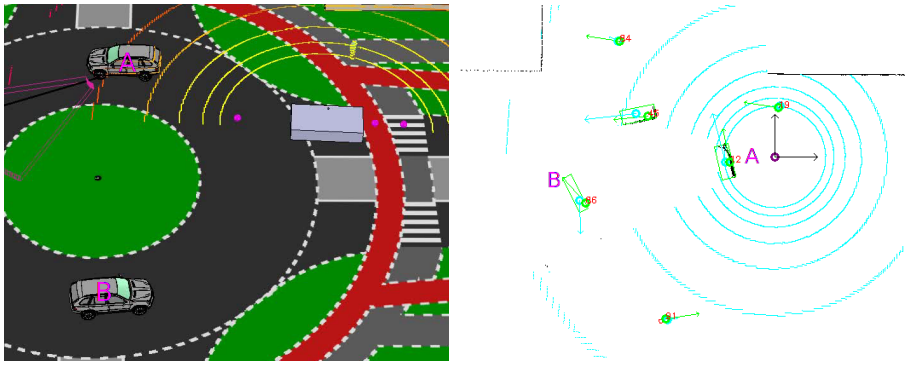
Figure 8.12: Roundabout Entrance 1



Einige Zeitschritte später sehen wir jedoch, dass unser Fahrzeug weiter in den Kreisverkehr reingefahren ist und dann wieder angehalten hat. Das liegt

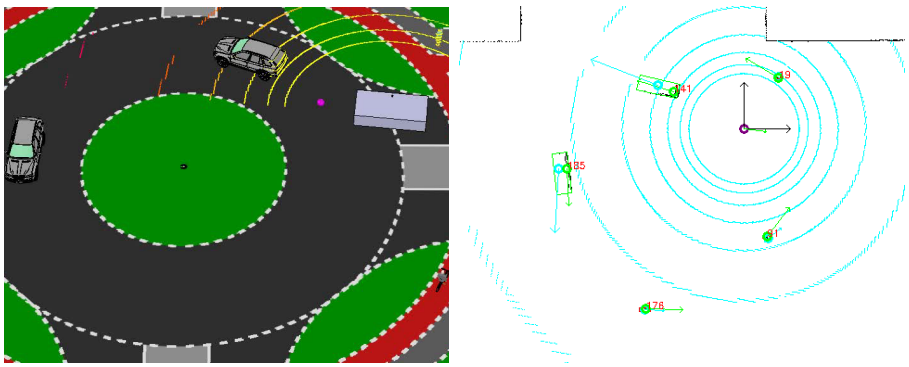
darin begründet, das Fahrzeug B kurz zuvor von Fahrzeug A verdeckt war und desshalb das Tracking versagt, wie in fig. 8.13 zusehen.

Figure 8.13: Roundabout Entrance 2



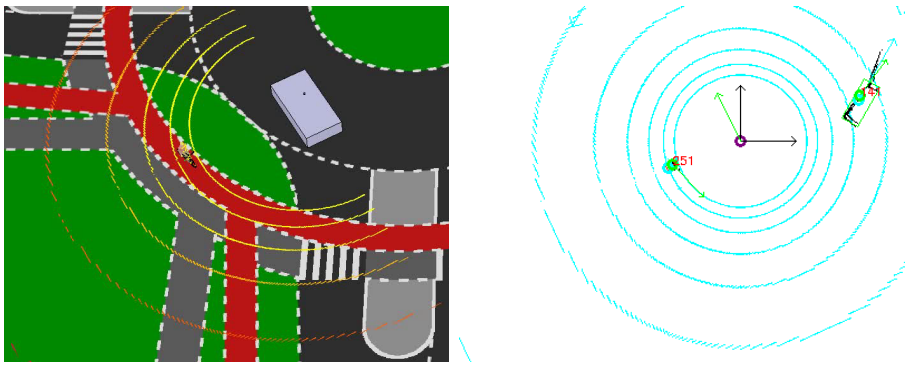
Diese Situation tritt noch ein weiteres mal auf, bis, wie in fig. 8.14 zu sehen, das Fahrzeug sicher in den Kreisverkehr einfahren kann.

Figure 8.14: Roundabout Entrance 3



Betrachten wir nun das herausfahren aus dem Kreisverkehr. In fig. 8.15 zu sehen, ist dass der Fahrzeug aus dem Kreisverkehr ausfahren will, jedoch vom Radfahrer daran gehindert wird.

Figure 8.15: Roundabout Exit 1



Dieser wird erfolgreich erkannt, sodass das Fahrzeug erfolgreich vor dem Radweg anhält. Zu sehen in fig. 8.16

Nachdem der Radwegwieder frei ist, kann das Fahrzeug den Kreisverkehr sicher verlassen, siehe fig. 8.17

Im weiteren Verlauf der Simulation treten immer wieder situationen auf, in dem das Auto aufgrund verdeckter Fahrzeuge den Kreisverkehr zu früh befährt. Dabei kahm es in keinem Fall zu einer Kollision. Jedoch ist dieser Zustand nicht optimal. Das Tracking kann die verdeckten Fahrzeuge meißt bis einen Zeitschritt bevor es wieder sichtbar wird verfolgen. Doch Der Konfidence wert des Objektes wird zu schnell zu niedrig und das Objekt wird verworfen. Weiterhin führt der Zeitraum in dem das Objekt am verschinden ist

Figure 8.16: Roundabout Exit 2

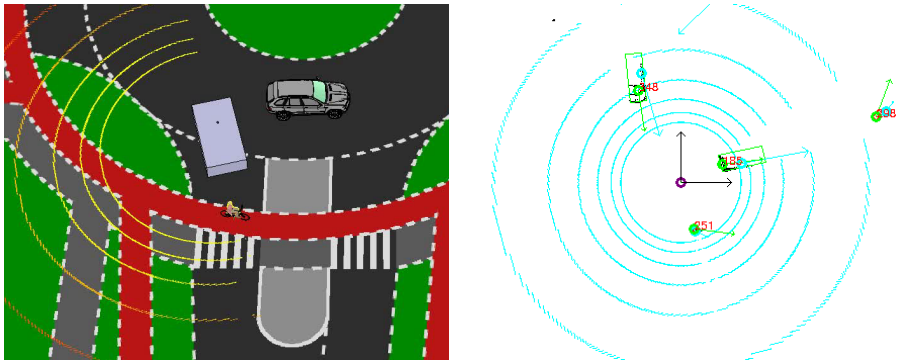
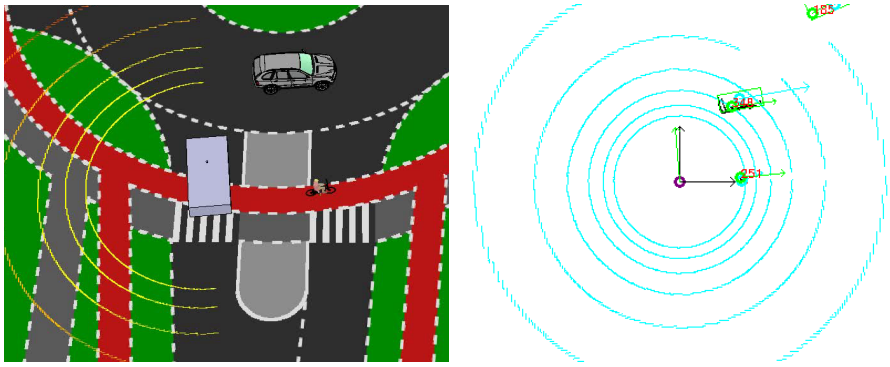


Figure 8.17: Roundabout Exit 3



zu einer falschen Rotation des Ojektes für wenige Zeitschritte, wie schon in fig. 8.6 zu sehen. Was dazu führt, dass die Berechnte Trajektorie des Trackings verfälscht wird und kein Optimales Tracking mehr erfolgen kann.

8.2 Real Measurements

Betrachten wir im folgenden Messdaten vom Realen Testfahrzeug “Snowfox” aud dem AstaZero Testgelände. Als Testumgebung wurde hier der Testraoundabout in fig. 4.1 gewählt. Daei handelt es sich um ein mini roundabout mit einem Durchmesser von 18m. Im rahmen des Scenarios fährt unser Fahrzeug von Norden in den Testkreisverkehr ein. In diesem Befindet sich bereits ein weiteres Fahrzeug, welches bei der Einfahrt beobachtet werden muss. Weiterhin befindet sich ein Fußgänger innehalb des Kreisverkehres, welcher bei der Ausfahrt beachtet werden muss.

8.2.1 Segmentation

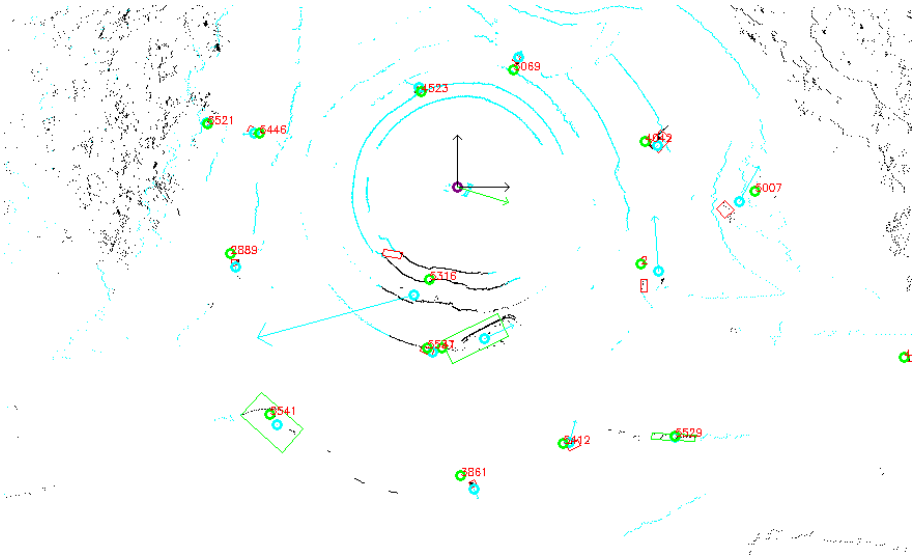
Im gegensatzt zur simulation erlauben es die Realen messdaten nicht, unsere Messwerte an Vergleichsdaten zu überprüfen. Die Messdaten sind jedoch unglich viel wertvoller, da sie keine Klinisch reine Umgebeung darstellen. Beispielsweise ist die Straße im gegensatzt zu Umgebung erhöht, diese ist dazu voller dicher Vegetation. Dies weiterhin ist unsere Straße nicht perfekt Plan, was eine weitere Herausforderung für die Segmentierung darstellt.

Zur evalierung der Segemntierung wurden alle Messframes von and begutachtet. Dabei wurden falsch positive Segementierungen der Straße notiert. Falsch positiv beideutet in diesem Fall, das Teile der Straße als nicht grund Klassifiziert wurden. Mehrere falsche erkennungen innerhalb eine Messframes wurden als ein Fehler gewertet. Es werden nur teile der Straße herangezogen, da alle anderen Bereich durch eine Karter herausgefiltert werden könnten. Weiterhin weil, eine bewertung per hand sonst nicht machbar währe. Zur bewertung der

False negative wurde nur das Auto begutachtet, da der Fußgänger auch als mensch in den Daten oft nicht zu erkennen ist. Damit eine Ekrennung als false Negative gewertet wird, müssen alle teile des Fahrzeuges als Boden klassifiziert werden. Es wurden insgesammt 819 Messframes Ausgewertet.

Die Auswertung der False positives hat 20 fälle egeben. Ein Beispiel dazu ist in fig. 8.18 zu sehen. Alle Hellblauen Bereich stellen dabei Boden dar, während schwarze bereiche potenzielle Objekte darstellen.

Figure 8.18: Car0 velocity



Es wurden in den Messframes keine False negativen gefunden. Die Gesammtanzahl der Fehler liegt somit bei 2.4%. In wie Fern sich die Fahlenden Messwerte des Sensors im hintern Bereich des Fahrzeuges auswirken lässt sich anhand der Daten leider nicht evalieren. Allerdings ist auch zu sagen das niemals ein Falspositive als Objekt erkannt wurde, das ist daruf zurückzuführen, dass dei False Positives immer nur für einen einzenen Frame vorhanden sind. Daher werden diese Durch den Confidence wert herausgefiltert und und würden das Fahrverhalten der Autonomen Fahrzeues nicht beeinträchtigen. Zu beachten ist allerdings auch, das die Segmentierung mit einem RANSAC abreitet, welcher nicht determinitisch ist. Daher kann dies bei einem weiteren durchlauf anders aussehen.

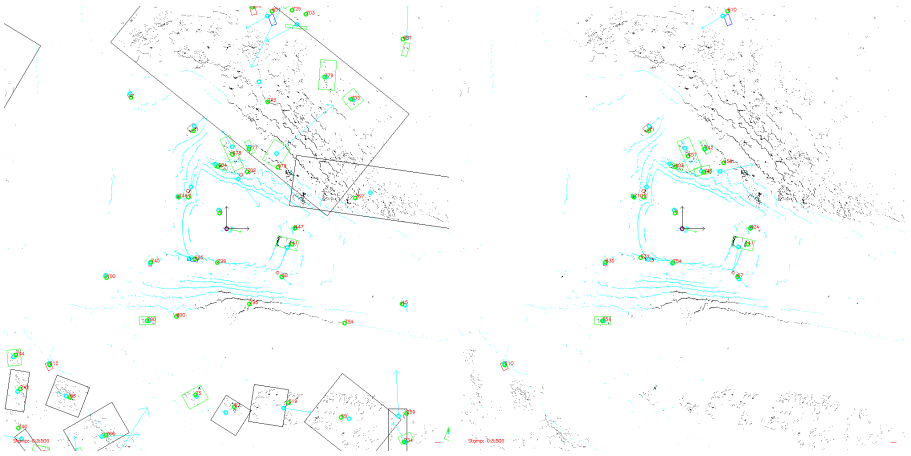
8.2.2 Confidence Filter

Weiterhin bieten uns die echten messwerte die Möglichkeit den Confiden Filter zu evaluieren. Da hier sehr viel Vegetation nicht als boden klassifiziert wird, was sich in geisterobjekten entlang des Straßenrandes auswirkt. Dazu werden im folgenden zwei momentaufnahmen betrachtet (fig. 8.19), einmal mit Filter und einmal ohne, um die qualität bewerten zu können.

Wie wir sehen können wird ein großteil aller ungültigen Onjekte herausgefiltert. Lediglich einige kleinere Objekte bleibe erhalten. Diese befinden sich jedoch stets außerhalb der Fahrbahn und ließen sich ebenfalls über eine Karte herausfiltern. Eine abschließende aussage lässt sich jedoch nur mit aufnahmen in einer echten Urbanen Umgebung, mit mehreren Fahrzeugen und andern Verkehrsteilnehmern machen

we and our enterfernen !! Überall!!!

Figure 8.19: Confidence Filtering

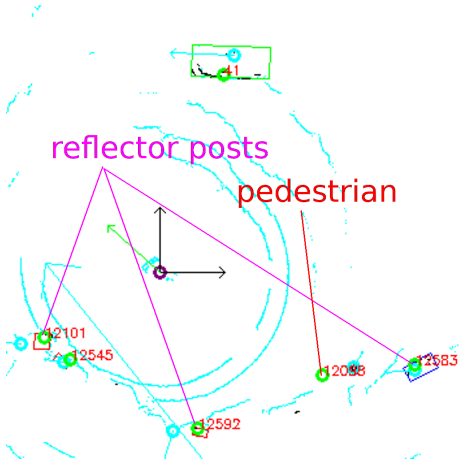


without filtering with filtering

8.2.3 Pedestrian

In diesem Abschnitt wird der Fußgänger im Scenario genauer untersucht. Dieser ist in diesem Szenarion besonder schwer zu erkennen, da der Kreisverkehr nicht über einen Fußweg verfügt und sich der Fußgängerdireckt am Sraßenrand befindet. Das Führt dazu das er für den Menschen aus der Vogelperspektive nicht von anderen Onjekten am Straßenrand, wie z.B. Leitpfosten fig. 8.20, zu unterschiden ist.

Figure 8.20: Pedestian and Reflector Posts



Problematisch ist dabei vorallem, das der Fußgänger nur durch seine Größe klassiffiziert wird. Dies führt dazu das er nicht von kleinen Statischen elementen zu unterscheiden ist. Befindet dieser sich nahe an einem statsichen Hindernis, kann es passieren, das diese als ein großes hinderiss erkannt werden. Das gleich würde auch bei größeren Menschenmassen an einem Fußgängerüberweg passieren. Dies ist in dieser Aufnahme jdeoch nicht passiert und der Fußgänger konnte zu jeder zeit erfolgreich als eigenständiges Objekt erkannt werden.

8.3 Performance

Abschließend gehen wir aud die laufzeit des Programmes ein dazu wurden Benchmarks auf zwei verschiedenen Systemen vorgenommen, die visualisierung der Messwerte wurde dabei deaktiviert. Dazu wurden insgesammt 893 Messframes ausgewertet und die bearbeitungszeit für jeden Frame gelogt. Das pro-

gramm wurde dazu mit folgenden Flags compiliert: “-march=native -O3 -flto -pipe” (GCC 6.3.1)

Dazu wurden folgende Testsysteme genutzt:

Desktop Computer

CPU AMD Phenom II X4 955 3.2 GHz

RAM 8GB DDR2 800

OS Arch Linux 4.9.16-1-lts

Notebook

CPU Intel Core i5-4210U 1.7 - 2.7 GHz

RAM 8GB DDR3 1333

OS Arch Linux 4.9.16-1-lts

Ergebnisse:

Desktop Computer

Mean 147.12 ms

Standard Deviation 32.3 ms

Notebook

Mean 30.7 ms

Standard Deviation 1.75ms

Da die Messerte des Velodyne Sensors mit 10Hz kommen, ist eine laufzeit von wneiger als 100ms nötig um die Daten in echtzeit auswerten zu können. Zu sehen ist das das ältere Desktop system diesen wert nicht erreicht. Das neuere Notebook hingegen erreich diesen Wert. Da sich das Fahrzeug jedoch mit zum teile hohen geschindigkeiten bewegt währe ein kleinerer wert wünschenswert um die Verzögerung der Messwerte zu reduzieren. Dabei sei Angemerkt, das es raum für optimierunen gibt. Da die reinen laufzeitwerte dazu wenig aussagekraft haben, wird im folgenden das Profiling chart (fig. 8.21) des Programmes betrachtet. Das Profiling cahrt wurde mit callgrind erstellt, welches die laufzeit über die Anzahl der Instuktionen abschätzt.

[1]. Es fällt auf, dass fast 40% der instruktionen auf die bereichsanfrage des DBSCAN fallen. kommen weitere Funktionen des DBSACN hinzu kommen erhöht sich dern anteil auf über 50%. Ein weiterer groteil der Laufzeit entfällt auf die kovertierung der Messwerte in kartesische kordinaten (6.1%) sowie Speicherverwaltung der Poincloud (23,61%). Also fallen über 75% der laufzeit auf wenig funktionlität des Programmes. Um das Programm zu optimieren sollte also eine Alternative für den DBSCAN gefunden werden. Wenn der Clusering algorythmus direckt auf den Polarkoordinaten arbeiten könnte fällt auch das Konvertieren der Pointcloud weg. Weiterhin ist es natürlich auch möglich den vorhandenen Algorythmus zu paralelisieren. Besonders für die Bereichsanfrage des DBSCAN ließe sich das einfach umsetzten.

1. `http://valgrind.org/docs/manual/cl-manual.html`

Figure 8.21: Pedestian and Reflector Posts

9 Conclusions

Kann in mehrere Unterkapitel gegliedert werden

Greift Thesen oder Fragestellungen aus der Einleitung wieder auf

Fasst die Arbeit knapp und prägnant zusammen

Ordnet die Ergebnisse in Gesamtzusammenhänge ein

Zieht Schlussfolgerungen aus den erarbeiteten Ergebnissen

Kann auch eigene Bewertungen oder Meinungen enthalten

Gibt eine Ausblick auf mögliche Konsequenzen oder notwendige weitere zu lösende Probleme

Bibliography

[1] Matthias Althoff and John M Dolan. “Online verification of automated road vehicles using reachability analysis”. In: *IEEE Transactions on Robotics* 30.4 (2014), pp. 903–918.

[2] Mohammed Aly et al. “Sensing , Navigation and Reasoning Technologies for the DARPA Urban Challenge 1 Introduction and Overview”. In: *Traffic* 1 (2007), pp. 1–25.

[3] A Atreya et al. “DARPA Urban Challenge – Virginia Tech Technical Paper”. In: (2007).

[4] R. Baumert. „Verkehrssicherheit und Leistungsfähigkeit von einstreifigen Kreisverkehrsplätzen - untersucht an Beispielen aus dem Kreis Borken“. Diss. Ruhr-Universität Bochum, 1998.

[5] Jon Louis Bentley. “Multidimensional binary search trees used for associative searching”. In: *Communications of the ACM* 18.9 (Sept. 1975), pp. 509–517. ISSN: 00010782.
DOI: 10.1145/361002.361007.
URL: http://portal.acm.org/citation.cfm?doid=361002.361007.

[6] Christian Berger. “Automating Acceptance Tests for Sensor- and Actuator-based Systems on the Example of Autonomous Vehicles”. PhD thesis. 2010, p. 298. ISBN: 9783832293789.
URL: http://www.christianberger.net/Ber10.pdf.

[7] Lothar Bondzio. *Verkehrssicherheit innerörtlicher Kreisverkehre*. Techn. Ber. Berlin: Gesamtverband der Deutschen Versicherungswirtschaft e. V., 2012.

[8] W Brilon, H Bäumer und O Flottmann. „Verkehrssicherheit an Mini-Kreisverkehren“. In: *Straßenverkehrstechnik* 46.4 (2002).

[9] Werner Brilon und Hanno Bäumer. *Überprüfung von Kreisverkehren mit zweistreifig markierter oder einstreifig markierter, aber zweistreifig befahrbarer Kreisfahrbahn*. 876. 2004.

[10] Manfred Broy. “Challenges in automotive software engineering”. In: *Proceeding of the 28th international conference on Software engineering - ICSE ’06*. New York, New York, USA: ACM Press, 2006, p. 33. ISBN: 1595933751.
DOI: 10.1145/1134285.1134292.
URL: http://portal.acm.org/citation.cfm?doid=1134285.1134292.

[11] Martin Buehler, Karl Iagnemma, and Sanjiv Singh. “The DARPA Urban Challenge”. In: *The DARPA Urban Challenge*. Vol. 49. 2010, p. 626. ISBN: 3642039901.
DOI: 10.1007/978-3-642-03991-1.
arXiv: arXiv:1011.1669v3.

[12] M Campbell and E Garcia. “Team Cornell: technical review of the DARPA urban challenge vehicle”. In: *DARPA Urban Chall. ...* (2007), pp. 1–25.
URL: http://archive.darpa.mil/grandchallenge/TechPapers/Team%7B%5C_%7DCornell.pdf.

[13] Oshkosh Truck Corp. “Team Terramax Darpa Grand Challenge 2005”. In: (2005), pp. 1–14.

[14] Applanix Corporation. *POSLV SPECIFICATIONS*. 2015.

[15] Martin Ester et al. “A density-based algorithm for discovering clusters in large spatial databases with noise”. In: AAAI Press, 1996, pp. 226–231.

[16] Martin a Fischler and Robert C Bolles. “Random Sample Consensus: A Paradigm for Model Fitting with Applicatlons to Image Analysis and Automated Cartography”. In: *Communications of the ACM* 24.6 (1981), pp. 381–395. ISSN: 0001-0782.
DOI: 10.1145/358669.358692.
URL: http://dx.doi.org/10.1145/358669.358692.

[17] Michael Galetzka and Patrick Glauner. “A Simple and Correct Even-Odd Algorithm for the Point-in-Polygon Problem for Complex Polygons”. In: *Proceedings of the 12th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*. SCITEPRESS - Science and Technology Publications, 2017, pp. 175–178. ISBN: 978-989-758-224-0.
DOI: 10.5220/0006040801750178.
URL: http://www.scitepress.org/DigitalLibrary/Link.aspx?doi=10.5220/0006040801750178.

[18] A S Goldberger. “Classical Linear Regression”. In: *Economic Theory* (1964), pp. 156–212.

[19] M Himmelsbach, T Luettel, and H.-J. Wuensche. “LIDAR-based 3D Object Perception”. In: *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems* (2009), pp. 994–1000.
DOI: 10.1109/IROS.2009.5354493.
URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5354493.

[20] M Holmes, a Gray, and C Isbell. “Fast SVD for large-scale matrices”. In: *Workshop on Efficient Machine Learning* 1.1 (2007), pp. 2–3.

[21] Alonzo Kelly. *A 3D state space formulation of a navigation Kalman filter for autonomous vehicles*. Pittsburgh: Carnegie Mellon University, 1994, p. 95.

[22] Jason Kong et al. "Kinematic and dynamic vehicle models for autonomous driving control design". In: *Intelligent Vehicles Symposium (IV), 2015 IEEE*. IEEE. 2015, pp. 1094–1099.

[23] James F. Kurose and Keith W. Ross. *Computer Networking A Top-Down Approach*. 5. 2013, p. 4. ISBN: 9780132856201.
DOI: 10.1017/CBO9781107415324.004.
arXiv: 8177588788.

[24] P. Lester. *Gang of Four*. Omnibus, 2008. ISBN: 9781847722454.
URL: https://books.google.de/books?id=9ApoM5GTpq0C.

[25] Bo Li, Tianlei Zhang, and Tian Xia. "Vehicle Detection from 3D Lidar Using Fully Convolutional Network". In: *Robotics: Science and Systems XII*. Robotics: Science and Systems Foundation, 2016. ISBN: 9780992374723.
DOI: 10.15607/RSS.2016.XII.042.
arXiv: 10.15607.
URL: http://www.roboticsproceedings.org/rss12/p42.pdf.

[26] Isaac Miller and Mark Campbell. "Rao-blackwellized particle filtering for mapping dynamic environments". In: *Robotics and Automation, 2007 IEEE International Conference on*. IEEE. 2007, pp. 3862–3869.

[27] Michael Montemerlo et al. "Junior: The stanford entry in the urban challenge". In: *Springer Tracts in Advanced Robotics* 56.October 2005 (2009), pp. 91–123. ISSN: 16107438.
DOI: 10.1007/978-3-642-03991-1_3.
arXiv: 10.1.1.91.5767.

[28] F. Moosmann, O. Pink, and C. Stiller. "Segmentation of 3D lidar data in non-flat urban environments using a local convexity criterion". In: *2009 IEEE Intelligent Vehicles Symposium*. June 2009, pp. 215–220.
DOI: 10.1109/IVS.2009.5164280.

[29] Abdul Nurunnabi, David Belton, and Geoff West. "Diagnostic-Robust Statistical Analysis for Local Surface Fitting in 3D Point Cloud Data". In: *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences* I-3.September (July 2012), pp. 269–274. ISSN: 2194-9050.
DOI: 10.5194/isprsannals-I-3-269-2012.
URL: http://www.isprs-ann-photogramm-remote-sens-spatial-inf-sci.net/I-3/269/2012/isprsannals-I-3-269-2012.pdf%20http://www.isprs-ann-photogramm-remote-sens-spatial-inf-sci.net/I-3/269/2012/.

[30] Meghashyam Panyam mohan Ram. "Least Squares Fitting of Analytic Primitives on a GPU". PhD thesis. Clemson University, 2007, p. 103.
URL: http://tigerprints.clemson.edu/all%7B%5C_%7Dtheses.

[31] For Immediate Release, Don Shipley, and Jan Walker. “Tartan racing wins $2 million prize for darpa urban challenge”. In: *Defense* March 2004 (2007), pp. 2004–2005.

[32] Maria Isabel Ribeiro. “Kalman and Extended Kalman Filters : Concept , Derivation and Properties”. In: *Institute for Systems and Robotics Lisboa Portugal* February (2004), p. 42.
DOI: 10.1.1.2.5088.
URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.2.5088%7B%5C&%7Drep=rep1%7B%5C&%7Dtype=pdf.

[33] Robin Schubert, Eric Richter, and Gerd Wanielik. “Comparison and evaluation of advanced motion models for vehicle tracking”. In: *Information Fusion, 2008 11th International Conference on* 1 (2008), pp. 1–6.
DOI: 10.1109/ICIF.2008.4632283.
URL: http://ieeexplore.ieee.org/xpls/abs%7B%5C_%7Dall.jsp?arnumber=4632283.

[34] Jarrod M Snider et al. “Automatic steering methods for autonomous automobile path tracking”. In: *Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RITR-09-08* (2009).

[35] Inge Söderkvist. *Using SVD for some fitting problems*. Tech. rep. 2. Sweden: Lulea University of Technology, 2009, pp. 2–5.
URL: https://www.ltu.se/cms%7B%5C_%7Dfs/1.51590!/svd-fitting.pdf.

[36] Forschungsgesellschaft für Straßen- und Verkehrswesen. Arbeitsgruppe Straßenentwurf. *Merkblatt für die Anlage von Kreisverkehren*. 2006.

[37] Forschungsgesellschaft für Straßen- und Verkehrswesen. Arbeitsgruppe Straßenentwurf. *Richtlinien für die Anlage von Landstrassen (RAL)*. FGSV (Series). FGSV-Verlag, 2013.

[38] Forschungsgesellschaft für Straßen- und Verkehrswesen. Arbeitsgruppe Straßenentwurf. *Richtlinien für die Anlage von Stadtstraßen: RASt 06*. FGSV (Series). FGSV-Verlag, 2007. ISBN: 9783939715214.

[39] Chris Urmson et al. “Tartan Racing: A Multi-Modal Approach to the DARPA Urban Challenge”. In: *Defense* 94.4 (2007), pp. 386–387. ISSN: 15564967.
DOI: 10.1002/rob.20251.
URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.107.8722%7B%5C&%7Drep=rep1%7B%5C&%7Dtype=pdf.

[40] Inc. Velodyne Acoustics. *Velodyne LiDAR Puck (VLP-16) Manual*. 2015.

[41] H Voß. „Zur Verkehrssicherheit innerörtlicher Knotenpunkte - Knotenpunkte mit Lichtsignalsteuerung, mit Vorfahrtregelung durch Verkehrszeichen, Kreisverkehrsplätze“. In: *Zeitschrift für Verkehrssicherheit* 40.2 (1994).

[42] Liang Zhang et al. “Multiple Vehicle-like Target Tracking Based on the Velodyne LiDAR*”. In: *IFAC Proceedings Volumes* 46.10 (June 2013), pp. 126–131. ISSN: 14746670.
DOI: 10.3182/20130626-3-AU-2035.00058.
URL: http://linkinghub.elsevier.com/retrieve/pii/S1474667015349211.