FULLY AUTOMATIC MULTI-LIDAR CALIBRATION FOR SELF-DRIVING CARS

Jörg P. Schäfer[1], Philipp Schmälzle[2], and Franz Andert[3]

[1]Inst. of Transportation Systems, *German Aerospace Center*, joerg.schaefer@dlr.de
[2]Inst. of Transportation Systems, *German Aerospace Center*, philipp.schmaelzle@dlr.de
[3]Inst. of Transportation Systems, *German Aerospace Center*, franz.andert@dlr.de

ABSTRACT

This paper addresses automatic vehicle-relative calibration of LiDAR sensors, which is essential for self-driving cars, e. g., for accurate localization, SLAM-based mapping, and sensor data fusion. While various semi-automatic techniques exist for the calibration of environment sensors, they still require a considerable amount of manual work or an expensive setup.

To that, this paper presents a new algorithm for finding the mounting poses of multiple low-density LiDAR-sensors on the vehicle while minimizing manual effort. In particular, the algorithm provides co-calibration as well as extrinsic calibration relative to the IMU installed on the vehicle. The calibration process only requires some recorded data from the sensors but no special environment to compute the accurate poses. In contrast to other algorithms that register point clouds to each other, this algorithm uses a newly developed map-to-map matching method increasing the robustness of optimization processes with regard to noisy data. It works by maximizing all sensors' consensus regarding a map simultaneously created from the recorded data.

We evaluated our algorithm using a car with six mounted LiDAR-sensors and recordings from different scenarios in real urban traffic and within a testing area. Our algorithm calibrates the sensors with an expected error of less than 5 cm and less than 0.5 degrees. A key contribution is its robustness against highly noisy initial sensor configurations and its accuracy regarding the angular calibration. A visualization of the recorded data also shows calibration improvements of the mounting poses over the manual measurements.

Keywords: LiDAR, Multi Sensor, Calibration, Autonomous Driving

1 INTRODUCTION

1.1 Motivation

The extrinsic calibration of LiDAR sensors on moving platforms such as self-driving cars is essential for further fusion and processing of the sensor data. In particular, the aggregation and evaluation of successive sensor data on such a platform requires the extrinsic calibration of the sensors to a localization solution such as GNSS. Various solutions exist to achieve this extrinsic calibration, some of which require heavy manual work, some of which require expensive sensors, and others require prerequisites such as perfect reference data to work properly.

1.2 Related Work

The obvious and probably most accurate method uses high quality measurement tools such as a tacheometer to measure the position of the LiDAR sensors. However, manually measuring the orientation of the sensors is a difficult task because of their small extent.

Semi-automatic technologies such as [3] detect dedicated objects simultaneously from different sensors. A post-processing algorithm estimates the relative poses of the sensors to each other by matching the detected object's pose from the different sensor's perspectives. However, this method does not find the absolute poses of the sensors on the platform.

Drones that systematically fly over an area and record stripes of it with an overlap use bundle adjustment [2,5] to match the recorded images in order to build a complete map of the recorded area. The theory behind bundle adjustment is also applicable to find the extrinsic calibration of the camera sensors on the drone.

The Boresight Calibration as in [4] is closest to our presented method. Assuming a static scene, the vehicle drives around this scene recording its position with a GNSS and the point clouds of the scene using a high density LiDAR sensor. A post-processing algorithm optimizes the pose of the LiDAR sensor in relation to the GNSS by matching these point clouds. The downside of this method is its requirement for expensive sensors, i. e., they require a high quality and high density LiDAR sensor.

1.3 Contribution

This work presents a new algorithm to automatically find the relative poses of multiple LiDAR sensors to a GNSS device on the vehicle. Knowing the absolute pose of the GNSS device on the vehicle even yields the absolute pose of the LiDAR sensors. Our algorithm shows high robustness even with low-density LiDAR sensors. On the downside, our current implementation only optimizes the pose within the horizontal plane.

So far, most approaches use scan matching or ICP-based[1] approaches to fit point clouds to each other. We contribute a new map-to-map matching method that would also be applicable for real-time applications such as SLAM.

2 METHODOLOGY

This section describes our algorithm that finds the relative pose of multiple LiDAR sensors to a GNSS device.

Our algorithm assumes recorded data from a coherent time and area consisting of point clouds measured by the LiDAR sensors as well as poses of the vehicle. The calibration process first converts each point cloud into a sensor map (cf. Figure 1) that contains probabilities of obstacles. Each sensor map is placed into the global space according to the global position of the car at the sensor map's time and the current pose of the sensor in the car. The overlap of the sensor maps is used to compute the inconsistency of the current sensor configuration, e. g., the variance of the observations at each global position. In each step, a gradient descent based optimizer changes the sensor configuration such that consistency of observations increases.

2.1 Sensor Maps

We consider a random variable X_ζ for each observed position ζ in the reference frame of the sensor. A value $X_\zeta = 1$ yields an obstacle at ζ. Hence, for a specific position ζ in the world, the probability that there is an obstacle at position ζ is $P[X_\zeta = 1]$. Now, each measurement O_i of the sensor's point cloud represents the measurement of an actual obstacle. Because of several uncertainties, this point represents the center of a probability distribution of the obstacle's actual place. For a measurement O_i, we assume a Gaussian distribution with a variance that is proportional to the point's distance to the sensor, i. e., $P[X_\zeta] \sim \mathcal{N}_{O_i, \rho \cdot \|O_i\|_2}$ where ρ is a constant value for fine-tuning the algorithm. In other words, the probability that $X_\zeta = 1$ increases with the proximity of a measurement O_i to ζ.

[1] The Iterative Closest Point (ICP) algorithm aligns two point clouds to each other.

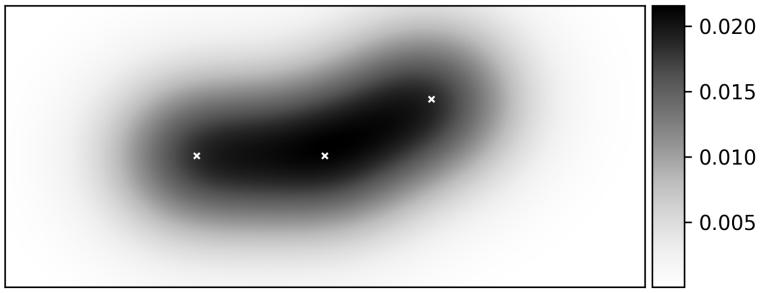


Figure 1: Sketch of a sensor map that shows the probability distribution of obstacles on the horizontal plane for a point cloud of three points.

For a point cloud with points $O_1, \ldots, O_n$, we consider each point O_i as a new measurement. Therefore, we have n random variables $X_{i,\zeta}$ at each position ζ that yield a complex probability distribution for the obstacles at each position. Given the current sensor data, the probability of an obstacle at position ζ is

$$P[Y_\zeta] := P\left[\bigvee_{i=1}^{n} X_{i,\zeta}\right] = 1 - \prod_{i=1}^{n} \left(1 - P[X_{i,\zeta}]\right).$$

Despite the information of an occurring obstacle, we can deduce even more information. When an obstacle occurs at position ζ, then there can be no (visible) obstacle between ζ and the origin of the sensor. Simply providing zero probability would yield ambiguous situations since the same information is provided for positions that the sensor didn't see at all. To that, we provide a weight W_ζ for the information $P[Y_\zeta]$ at each position by integrating $P[Y_{\zeta^*}]$ along the line through the sensor's origin and ζ from ζ to infinity:

$$W_\zeta = \int_{\zeta}^{\infty} P[Y_{\zeta^*}] d\zeta^*$$

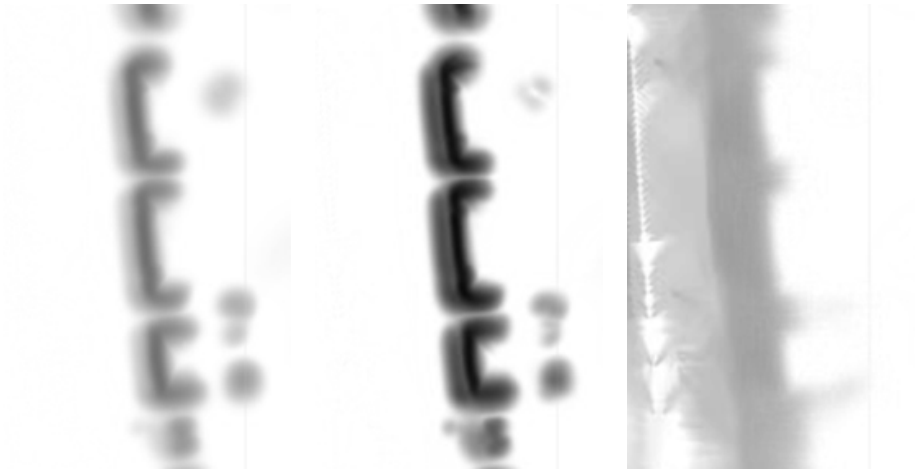
The intuition behind the weights is the following: Assume a Monte Carlo simulation of the obstacles distributed via $P[Y_\zeta]$. Every time, the obstacle occurs at or behind ζ, we can deduce that there is no obstacle in front of ζ. The relative frequency of these situations is W_ζ.

Now, a sensor map holds both, the probability $P[Y_\zeta]$ and W_ζ at each position ζ. In practical implementations, the sensor maps are implemented as grid maps and the probability distributions are considered in bounded areas only.

2.2 Global Variance Map

The sensor maps are placed in a global reference frame according to the vehicle's pose at the time of recording and the pose of the sensor in the vehicle. Due to overlap of the globally placed sensor maps, we obtain multiple (weighted) measurements at each global position ξ. Hence, aggregating the sensor maps yields an expected value, a variance, and a summed weight at each position ξ, respectively.[2] A high variance (especially with a high weight) at a position ξ yields an inconsistent sensor configuration. Hence, the goal is to minimize these variances, considered as inconsistencies. Figure 2 shows an example for a global map created from a recording in real urban traffic.

[2]When no measurement was provided for a position ξ, we assume the expected value, variance, and weight to be 0.



(a) Expected value (b) Variance (c) Weight

Figure 2: An example for a global map that shows parking cars recorded from misconfigured sensors and at successive points in time.

2.3 Global Likelihood

The *likelihood map* aggregates the likelihood of all sensors being correct at each position in space. To that, consider a position ξ and all sensor maps overlapping ξ. We denote the sensor map values $P[Y_\xi]$ of the i-th sensor map with $P[Y_{i,\xi}]$.

Now, either there is an obstacle at ξ or there is not. If there is an obstacle, then the probability of all measured values to agree is

$$P[Y_{1,\xi} \wedge \ldots \wedge Y_{n,\xi}] = \prod_{i=1}^{n} P[Y_{i,\xi}].$$

If, on the other hand, there is no obstacle at ξ then the probability of all measured values to agree is

$$P[\neg Y_{1,\xi} \wedge \ldots \wedge \neg Y_{n,\xi}] = \prod_{i=1}^{n} \left(1 - P[Y_{i,\xi}]\right).$$

The overall probability of the sensors to agree is

$$P[Z_\xi] = \prod_{i=1}^{n} P[Y_{i,\xi}] + \prod_{i=1}^{n} \left(1 - P[Y_{i,\xi}]\right). \tag{1}$$

To take the limited visibility of the world into account, we need to consider the weight $W_{i,\xi}$ of each sensor's measurements. A simple approach is to use the weight as linear interpolation factor introduced to the product, e. g., when the weight is 0 then the measurement does not change anything (i. e., we multiply by 1) and when the weight is 1 it provides a full change to

the overall probability of the sensors to agree (cf. Equation (1)). This changes (1) as follows:

$$\begin{aligned} P[Z_\xi] = &\prod_{i=1}^{n} \Big(W_{i,\xi} \cdot P[Y_{i,\xi}] + (1 - W_{i,\xi}) \cdot 1 \Big) \\ &+ \prod_{i=1}^{n} \Big(W_{i,\xi} \cdot (1 - P[Y_{i,\xi}]) + (1 - W_{i,\xi}) \cdot 1 \Big) \\ = &\prod_{i=1}^{n} \Big(W_{i,\xi} \cdot P[Y_{i,\xi}] + 1 - W_{i,\xi} \Big) \\ &+ \prod_{i=1}^{n} \Big(1 - W_{i,\xi} \cdot P[Y_{i,\xi}] \Big). \end{aligned}$$

If there is no measurement for a position ξ, then $P[Z_\xi] = 1$.

The probability of all sensors to agree at every position ξ in space is

$$\mathcal{L} = \prod_{\xi} P[Z_\xi].$$

Since the optimizer needs the derivative, which is expensive to compute for a product, we consider the log-Likelihood that has its maximum at the same parameters

$$\log \mathcal{L} = \log \prod_{\xi} P[Z_\xi] = \sum_{\xi} \log P[Z_\xi].$$

Note, that taking the logarithm is well-defined since the values $P[Z_\xi]$ will always be positive. To that, recall that $P[Z_\xi]$ contains a product of $P[\neg Y_{i,\xi}]$ that are all positive since no sensor is a 100% certain of a measurement, i. e., all $P[Y_{i,\xi}]$ are less than 1.

An optimizer that tries to find a minimum simply needs to negate this value, i. e., the likelihood map should contain $-\mathcal{L}$ as inconsistency values.

2.4 Optimizer

We assume that the localization solution of the vehicle (e. g., a GNSS with an integrated inertial measurement unit) provides accurate relative movements or positions. We further assume that the error of the sensor's measurements is relatively small compared to the error of the sensor configuration.

Integrating the inconsistency values over the global space provides a target function that can be minimized by an optimizer. We use a gradient descent based optimizer with momentum to overcome local minima. We also experimented with the sophisticated optimizer Ceres [1], which however did not perform as well as the gradient descent based optimizer.

2.5 Towards Localization

The global map created by the optimizer provides statistical information about obstacles in the surrounding area. Assuming that a GNSS such as GPS works accurately in average, we can assume that multiple recordings of the same area yield an accurate map. This global map enables scan-matching and map-to-map matching based localization strategies.

To localize the vehicle in global space, consider the likelihood of the current sensor map given the prior of the global map. The computation is similar to the computation of the likelihood map , but instead of multiple sensor maps we now only consider the current sensor map and the

(a) Line Slalom Box-Targets

(b) Zig Zag Slalom L-Targets

Figure 3: Airfield proving ground layout examples

global map. Alternatively, the variance between the current sensor map and the global variance map yields a localization solution. The difference to our calibration solution lies in the optimizer that estimates the current pose of the vehicle instead of the pose of its sensors.

3 EVALUATION

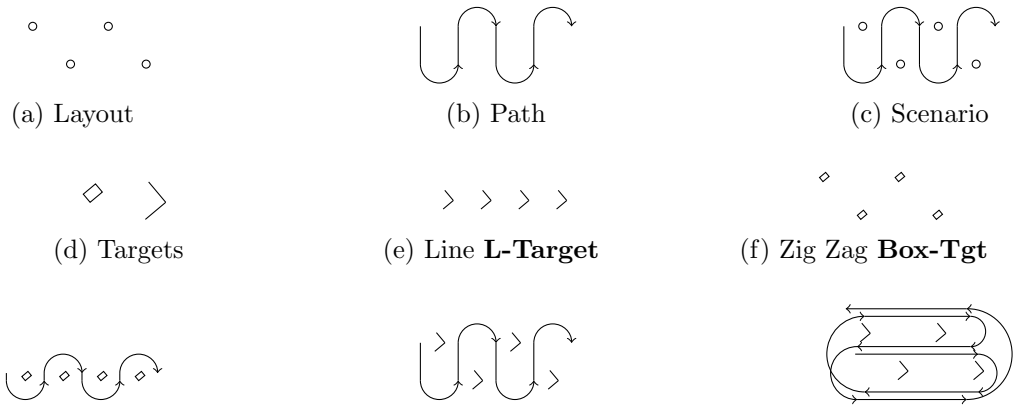
The evaluation of our method reveals partially surprising and paradox results, as we will show in this section. We evaluate our method on data recorded during different measurement scenarios with our research vehicle. Since the results the global likelihood method showed no convergent behavior, we skip it in this section. The evaluation is done with respect to the influence of layout, scenario, target, and initial sensor configuration. We also evaluate the results visually directly on the recorded data. The section concludes with a discussion.

3.1 Experimental Setup

Scenario Description As proving ground we used a small parking lot and a research airfield. The parking lot has water drainage slopes in the driving path, whereas the airfield provides a wide and flat area with a $\approx$ 25m wide lane and a grass road edge. Figure 3 shows the airfield with two example layouts. Figure 4 visualizes the terminology and experimental setup for the evaluation. In particular, Figure 4d shows the two types of *target* shapes (*Box* and *L*) to be detected by the sensors. The *layout* (cf. Figure 4a) describes the pattern in that the targets are placed. The *path* (cf. Figure 4b) declares the driving track through the targets. A *scenario* (4c) combines both, a layout and a path. Box targets are two cardboard boxes stacked on top of each other, whereas L-targets cardboards taped together such that they form an L-shape. Figure 4e and 4f represent the two layouts used in our experiments. The longitudinal distance between two boxes is $\approx$ 10m. With the Zig Zag layout, the lateral distance is $\approx$ 7.5m. Figure 4g, 4h and 4i visualize the scenarios that we drove during our experiments. In total, we recorded 26 drives at the airfield and seven drives at the parking lot on 14 different scenarios[3].

Vehicle Description Our research vehicle is a (self-driving) VW Passat GTE 2016 fitted with multiple sensors, e. g., six IBEO Lux 4L LiDAR sensors, each of which has a horizontal field of view of $+50°$ to $-60°$, four layers with 0.8° vertical resolution, and 0.25° horizontal resolution. We recorded the LiDAR data with 25Hz. Our GNSS system is a NovAtel ProPak6 with a μIMU-IC as inertial measurement unit with differential GPS correction disabled. The sensor heads are mounted 30 cm above the ground with 0° pitch and roll angle.

[3]Figure 4g: Tgt=Box/L, Start=east/west, and mirrored at the east-west axis; Figure 4h: Tgt=Box/L, Start=east/west; Figure 4i: Tgt=Box/L



(a) Layout (b) Path (c) Scenario

(d) Targets (e) Line **L-Target** (f) Zig Zag **Box-Tgt**

(g) Line Slalom (h) Zig Zag Slalom (i) Zig Zag around

Figure 4: Experiment terminology

Hyperparameters We used the same optimization parameter set for all runs in our evaluation. We clipped the LiDAR sensor data to a range of 0.5m to 20m. For our algorithm, we rastered the global space to a grid with a resolution of 2cm and fixed the parameter $\rho = 0.1$ (cf. section 2). The optimizer always performed 100 iterations per run.

3.2 Robustness

We combined different recordings to form the following *runs*: The results of this evaluation are represented by multiple *runs*.

#1: all 26 drives from the airfield

#2: all recordings with slalom paths grouped by layout and target type

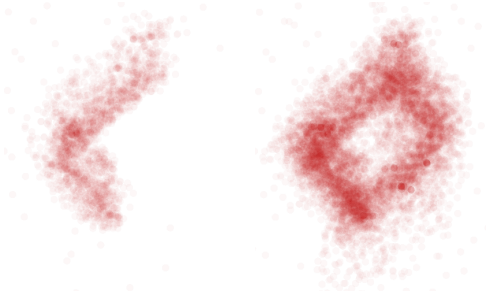
#3: all seven drives on the parking lot

#4: identical scenarios as in #2 but with very inaccurate initial starting pose of all LiDAR sensors

Description Table 1 Table 1 shows the mean of the second and third quartile of all calibration results within each run. Furthermore, it shows the interquartile range (IQR) that corresponds to the variation within the data points.

Description Figure 6 Figure 6 shows the behavior and results of the calibration processes. The initial starting position is marked with a dot. The Cartesian coordinate system shows the optimization of the X and Y position of the sensors, whereas the polar coordinate plot shows the yaw angle. The origin of the Cartesian coordinate system marks the position of the IMU.

Evaluation Box vs. L Target Shape Comparing runs with Box vs L targets shows the influence of the target shape and expansion. In most cases, the IQR of runs with box-shaped targets is significantly higher, which is caused by a higher number of outliers when using the box-shaped targets. As Figure 6c and 6g show, the optimization process diverges more often on box-shaped targets, in particular regarding the rear center sensor (green). To compensate this misplacement, those runs show high angular miscalibration of the rear center sensor as well. The proposed algorithm seams to work better if the targets are thin walled, which addresses the possibility of matching the left/front side of the box with its right/back side.



(a) Run #1 (b) Run #2

Figure 5: Comparison of Run #1 and #2: Recorded point clouds of a box target from the front left sensor on a line slalom scenario.

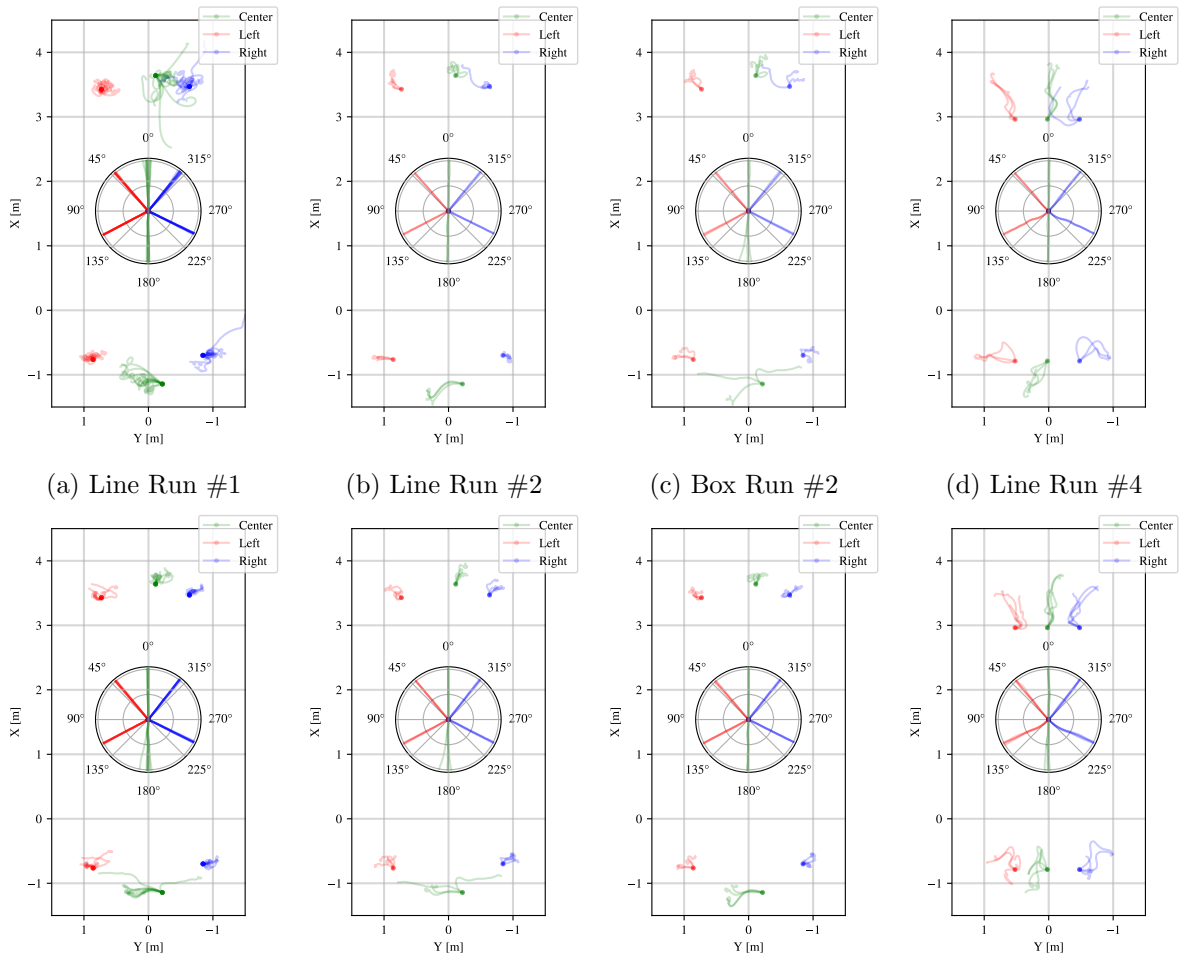
Evaluation Target Coverage Run #2 contains the same data points as run #1 but with multiple scenarios merged to a batch as if the recordings were one scenario. We used this approach to get scenarios that provide a higher grade of target coverage for each sensor. For example, in the scenario shown in Figure 4g, the front left sensor observes only a small area of each box. Merging four drives with four different paths enables the same sensor to cover nearly 360° of all targets. Figure 5 shows the increased target coverage when merging multiple scenarios.

Evaluation of the Layout and Target Coverage To evaluate the impact of the layout on the quality of the calibration, we compared the results of all slalom drives on line layouts with all slalom drives on zig zag layouts (cf. Table 1). The convergence paths as well as the results of the zig zag layout show similar behavior in run #1 and #2 (cf. Figure 6e and 6f), whereas the line layout (cf. Figure 6a and 6b) diverges much more and shows higher number of outliers. These results indicate the disadvantage of the line layout and its little target coverage. Merging these drives increases the target coverage and therefore stabilizes the optimization. On the other hand, the target coverage of the zig zag layout seems well enough already; thus, there is only little improvement from run #1 to run #2.

Evaluation of Initial Configuration The comparison of run #2 and run #4 shows the qualitative influence of the initial starting position. Both set of optimization processes use the same set of merged scenarios and the same hyperparameters. Figure 6h shows that the calibration process still converges, even though the initial starting position is chosen by good guess. Comparing Figure 6b and 6f with Figure 6d and 6h shows the influence of the initial configuration on the final results, respectively. In particular, Figure 6b and 6d show the same convergence points though starting with different initial configurations.

Visual Evaluation To visually evaluate the calibration results, we aggregate all point clouds of all sensors regarding each new sensor configuration. We observe the targets and the sharpness of its boundaries in the resulting aggregated point cloud. In particular, Figure 7 shows one L target as it is seen with the ground truth calibration as well as by the sensor configurations that result from different calibration processes. Except for the results of run #3, all new sensor configurations yield sharper observed targets compared to the original sensor configuration.

[4]Ground truth was measured with wrong sign



(a) Line Run #1 (b) Line Run #2 (c) Box Run #2 (d) Line Run #4

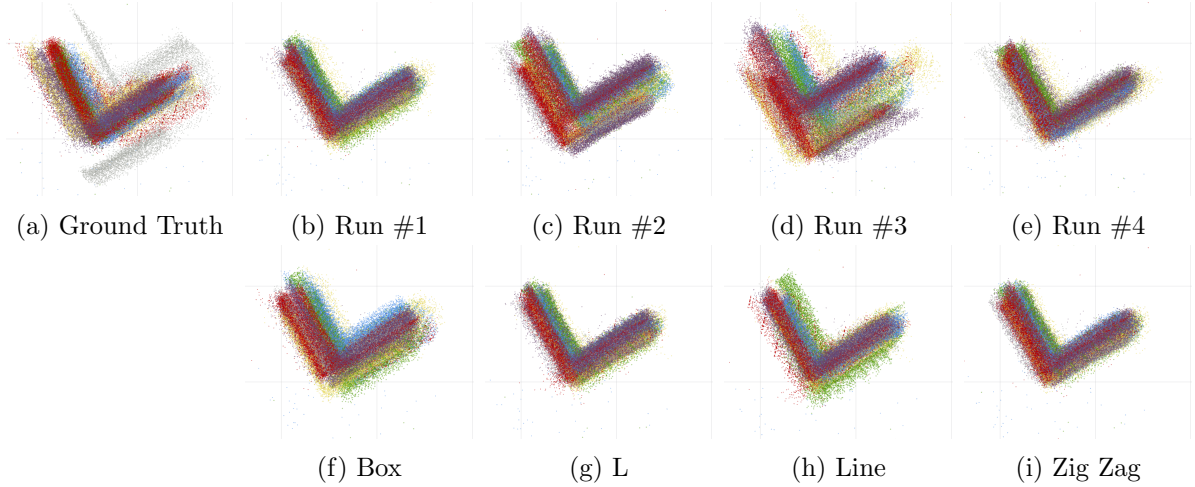
(e) Zig Zag Run #1 (f) Zig Zag Run #2 (g) LBox Run #2 (h) Zig Zag Run #4

Figure 6: Comparing optimization progress (100 iterations) form line and zig zag layout of run #1, #2, and #4. The paths within the circles describe the yaw angles of the sensor configurations during the optimization process from the first iteration (center) to the last iteration (outer ring). The outermost paths of each rectangle describe the X-Y-coordinates of the sensor configurations during the optimization process. The coordinate system is centered at the origin of the IMU; X and Y point to the driving direction and to the left, respectively.

Table 1: LiDAR position and orientation results of evaluation runs. Green background marks closest value to ground truth.

		front								
		left			***center***			***right***		
		x	y	Ψ	x	y	Ψ	x	y	Ψ
#1	*abs*	3.49	0.8	0.7054	3.7	-0.23	6.2907	3.54	-0.69	5.6147
	IQR	0.06	0.16	0.0179	0.19	0.2	0.026	0.05	0.2	0.016
#2	*abs*	3.54	0.88	0.7075	3.79	-0.16	6.2796	3.58	-0.71	5.603
	IQR	0.06	0.07	0.0194	0.08	0.12	0.0124	0.1	0.15	0.019
#3	*abs*	3.45	1.05	0.6931	3.72	-0.32	6.3014	3.49	-0.85	5.6214
	IQR	0.04	0.05	0.0092	0.05	0.19	0.0121	0.03	0.06	0.0058
#4	*abs*	3.5	0.81	0.7063	3.75	-0.09	6.2816	3.49	-0.56	5.6124
	IQR	0.08	0.21	0.0206	0.12	0.13	0.011	0.08	0.25	0.0134
Box	*abs*	3.53	0.79	0.6978	3.67	-0.3	6.2935	3.57	-0.81	5.6144
	IQR	0.07	0.27	0.0118	0.27	0.15	0.0331	0.05	0.17	0.0118
L	*abs*	3.47	0.8	0.7093	3.71	-0.19	6.2894	3.53	-0.67	5.6142
	IQR	0.05	0.09	0.0131	0.11	0.17	0.0208	0.04	0.1	0.019
Line	*abs*	3.48	0.7149	0.71	3.57	-0.28	6.2947	3.55	-0.61	5.5994
	IQR	0.08	0.13	0.0135	0.22	0.2	0.056	0.05	0.18	0.0426
ZigZag	*abs*	3.49	0.83	0.703	3.77	-0.15	6.2876	3.53	-0.69	5.6184
	IQR	0.1	0.03	0.064	0.03	0.14	0.0176	0.05	0.15	0.0075
GT	*abs*	3.43	0.73	0.6861	3.64	-0.11	0.00	3.47	-0.64	5.6025

		rear								
		left			***center***			***right***		
		x	y	Ψ	x	y	Ψ	x	y	Ψ
#1	*abs*	-0.72	0.95	2.0569	-1.11	0.28	3.1384	-0.67	-0.98	4.2543
	IQR	0.04	0.12	0.0048	0.15	0.2	0.0218	0.03	0.15	0.0127
#2	*abs*	-0.71	1.08	2.0561	-1.14	0.29	3.124	-0.65	-0.97	4.2498
	IQR	0.08	0.11	0.0098	0.25	0.06	0.0113	0.16	0.05	0.0127
#3	*abs*	-0.77	1.12	2.0688	-1.15	0.54	3.1536	-0.71	-1.12	4.2359
	IQR	0.03	0.03	0.011	0.05	0.02	0.0036	0.03	0.04	0.0121
#4	*abs*	-0.71	0.95	2.0546	-1.14	0.32	3.128	-0.67	-0.92	4.2482
	IQR	0.08	0.32	0.01	0.32	0.02	0.0102	0.16	0.19	0.0129
Box	*abs*	-0.7	0.97	2.0585	-1.0	0.28	3.1415	-0.64	-1.05	4.2453
	IQR	0.07	0.24	0.0056	0.28	0.36	0.0387	0.03	0.11	0.0065
L	*abs*	-0.73	0.96	2.0562	-1.13	0.28	3.1373	-0.67	-0.95	4.2582
	IQR	0.04	0.03	0.004	0.13	0.15	0.0183	0.02	0.08	0.0088
Line	*abs*	-0.72	0.91	2.0567	-1.04	0.21	3.1425	-0.68	-0.98	4.2593
	IQR	0.05	0.13	0.0044	0.17	0.23	0.0198	0.05	0.12	0.0082
ZigZag	*abs*	-0.71	0.96	2.0565	-1.12	0.31	3.1348	-0.66	-0.94	4.2522
	IQR	0.04	0.02	0.0041	0.1	0.04	0.0185	0.02	0.14	0.0125
GT	*abs*	-0.76	0.86	2.0682	-1.14	-0.22^4	3.1415	-0.7	-0.84	4.2412



(a) Ground Truth (b) Run #1 (c) Run #2 (d) Run #3 (e) Run #4

(f) Box (g) L (h) Line (i) Zig Zag

Figure 7: Comparing global point cloud location of the same measurement drive with different LiDAR calibrations. Grid size of 1 m. Each sensor has an individual color. 7f–7i are subsets of run #1

3.3 Discussion

The most obvious and encouraging proof of concept is the lateral position correction of the rear center sensor: Here, the initial position had been stored with a wrong sign and therefore on the wrong side of the vehicle. Our calibration method relocated the sensor to the correct side. Our proposed method proves to be robust against several initial displacements that enables its use without any prior knowledge of the exact mounting position.

Figure 7 shows the improvement of the observations by LiDAR sensors with recalibrated sensors. Additionally, taking subsection 3.2 into account, the best results should be achieved with thin L targets in a line layout with increased target coverage. The drives should be repeated several times to use simple clustering methods to filter outliers.

While this calibration process yields sharper observed objects and therefore seems to be rather correct, Figure 8 shows the position w.r.t. the vehicle which is impossible by definition: The calibration process places the LiDAR sensors outside of the vehicle shape and therefore into a clearly implausible mounting position[5]. We assume that this is causes by errors (i. e., delays) within the used signals representing the vehicle dynamics. Such delays cause the measured sensor to target distance to decrease when the car accelerates forward, whereas the GPS pose remains in its initial state of still standing. The same behavior occurs during yaw movements. Such behavior leads to a systematical assumption of too short distances. The optimizer addresses this effect by placing the sensors further away from the IMU. Both, the inaccuracies of the LiDAR sensors itself and the windy weather conditions at the time of recording might cause the wide expansion of the rather thin L shaped target. Furthermore, the optimization process during our evaluation probably did not reach a global minimum but has either stopped in a local minimum or was forced to stop by reaching the 100^{th} iteration.

4 CONCLUSION AND FUTURE WORK

In this work, we present a novel algorithm for the extrinsic calibration of LiDAR-sensors on a vehicle with minimal manual effort. We evaluate the algorithm in different settings and show that it delivers a solution with high accuracy. The algorithm proved to be robust even when the initial sensor configuration is a rough guess only. Still, we hope to improve the quality of

[5]We mounted the sensors within the bumpers and therefore within the shape shown in Figure 8.

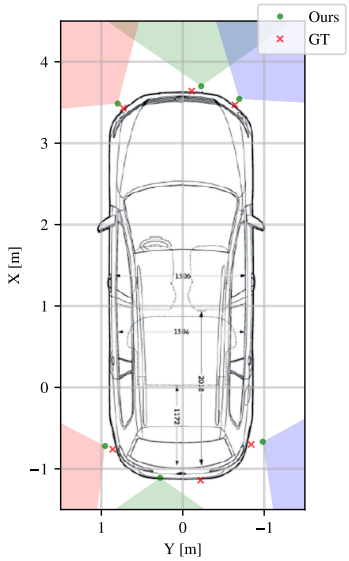


Figure 8: Final sensor mount position measured ground truth (GT) compared to our method (run #1) with sensor field of view

our calibration process by improving the signal quality and time synchronization of the LiDAR sensors and our GNSS/IMU.

Our algorithm is based on a novel map-to-map matching strategy, which yields a proof-of-concept to this new strategy. We further propose a map-to-map matching based algorithm for the localization problem that follows the same algorithmic idea as the calibration algorithm. The implementation of this localization algorithm as well as a documentation to the algorithmic background enabling us to compute map-to-map matching results in real-time are already in progress.

REFERENCES

[1] Sameer Agarwal, Keir Mierle, and Others. Ceres solver. `http://ceres-solver.org`.

[2] Manolis I. A. Lourakis and Antonis A. Argyros. Sba: A software package for generic sparse bundle adjustment. *ACM Trans. Math. Softw.*, 36(1), March 2009.

[3] Zoltán Pusztai, Iván Eichhardt, and Levente Hajder. Accurate calibration of multi-lidar-multi-camera systems. *Sensors*, 18(7), 2018.

[4] P. Rieger, Nikolaus Studnicka, M. Pfennigbauer, and G. Zach. Boresight alignment method for mobile laser scanning systems. *Journal of Applied Geodesy*, 4:13–21, 01 2010.

[5] Bill Triggs, Philip F. McLauchlan, Richard I. Hartley, and Andrew W. Fitzgibbon. Bundle adjustment — a modern synthesis. In Bill Triggs, Andrew Zisserman, and Richard Szeliski, editors, *Vision Algorithms: Theory and Practice*, page 298–372, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg.