Fully Automatic Multi-LiDAR Calibration For Self-Driving Cars

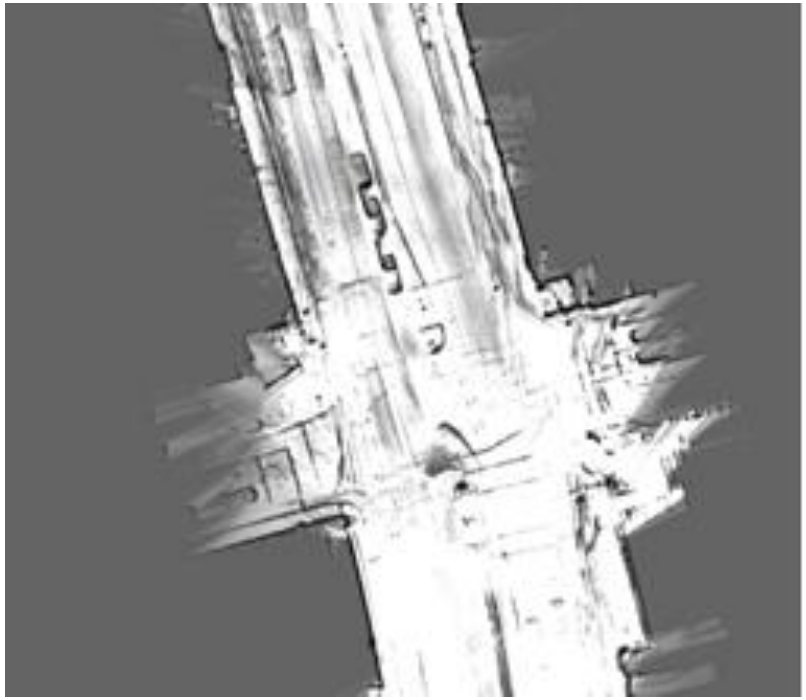
Jörg P. Schäfer, Philipp Schmälzle, Franz Andert
DLR

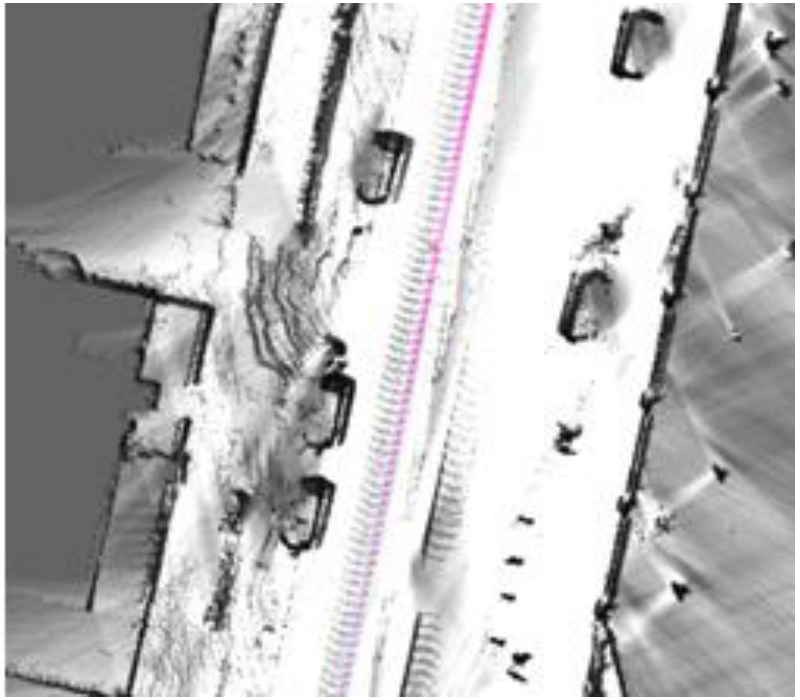
Outline

- Introduction
- Methodology
- Experimental Results
- Future Work

Motivation

- An autonomously driving car drive as good as its perception is
  - Localization
  - Object Detection

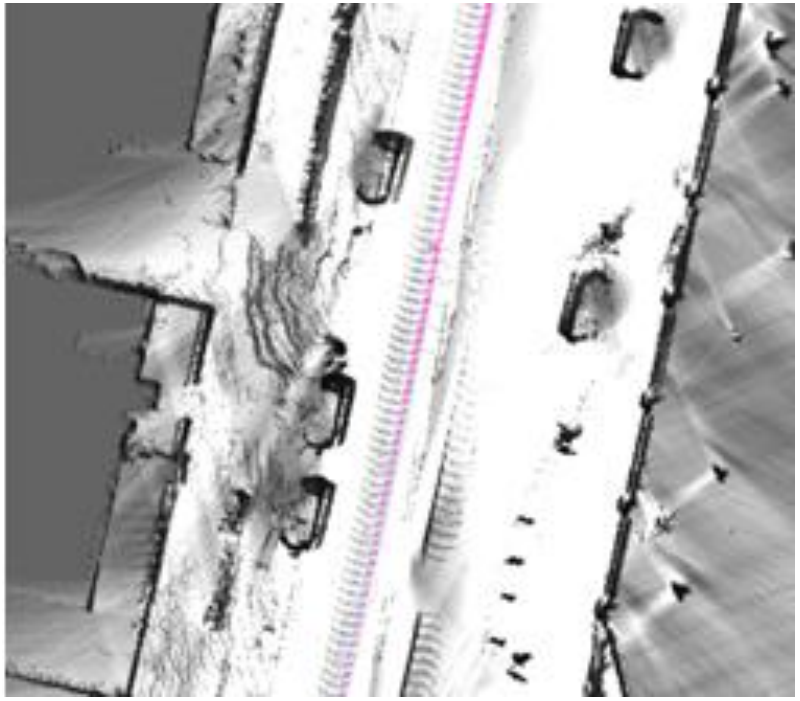




Made with goole cartographer; W. Hess, D. Kohler, H. Rapp, and D. Andor, Real-Time Loop Closure in 2D LIDAR SLAM, in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE, 2016. pp. 1271–1278.

Motivation

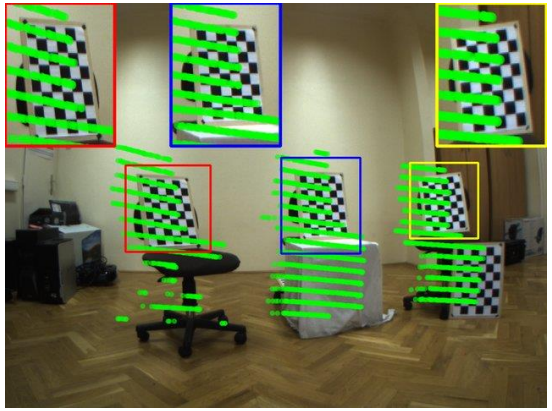
- An autonomously driving car drives as good as its perception
  - Localization
  - Object Detection
- Inconsistencies of calibration yields inaccurate results
  - Optimization process for localization often even doesn‘t converge
  - Object Detection yields inaccurate positions
- Our case: LiDAR sensor position and orientation measured with total station
  - Orientation inaccurate
  - Manual errors on position
- **Goal**: avoid these errors

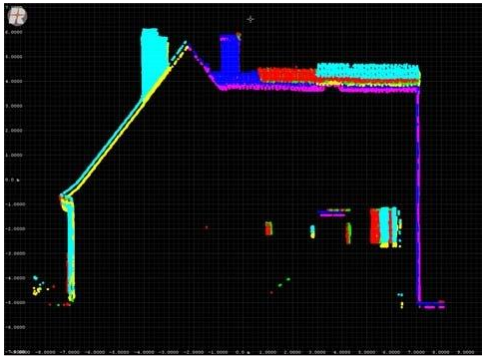


Made with goole cartographer; W. Hess, D. Kohler, H. Rapp, and D. Andor, Real-Time Loop Closure in 2D LIDAR SLAM, in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE, 2016. pp. 1271–1278.

Automatic Calibration – Related Work

- Automatically calibrate sensors based on consistency of perceived data!
- Z. Pusztai, I. Eichhardt, and L. Hajder propose a method based on detecting boxes from different sensors (cameras and LiDAR sensors)
  - No calibration relative to the plattform
  - Specific environment necessary
- P. Rieger, N. Studnicka, and M. Pfennigbauer use Boresight Calibration to align a LiDAR sensor relative to a IMU/GNSS
  - Close to our approach, but requires high quality / high density sensors



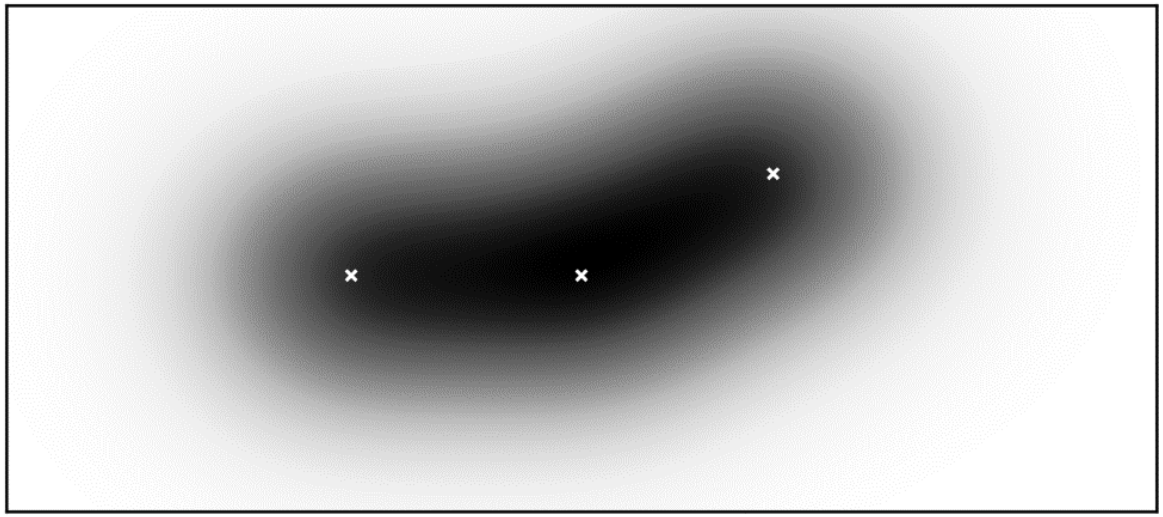


Outline

- Introduction
- Methodology
- Experimental Results
- Future Work

Sensor Maps

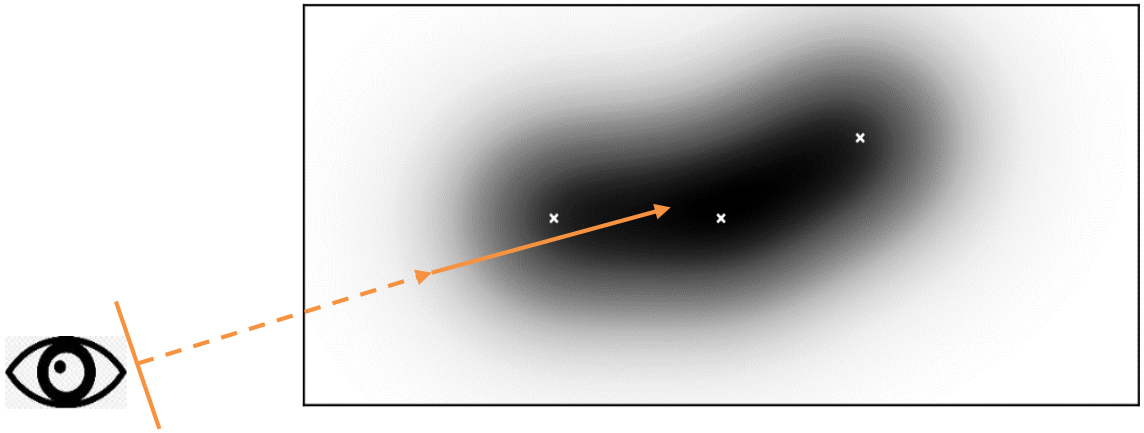
- Low-density sensors provide less information (our case: 4 layers)
- Increase information using stochastic approach
- Points O_i of a sensor measurement represent surface points
- Gaussian Distribution around each O_i to estimate surface points nearby
    - Yields random Boolean variables $X_{\varsigma,i}$ at each position ς
- Aggregate $Y_\varsigma = \bigvee_i X_{\varsigma,i}$



Sensor Map Weights

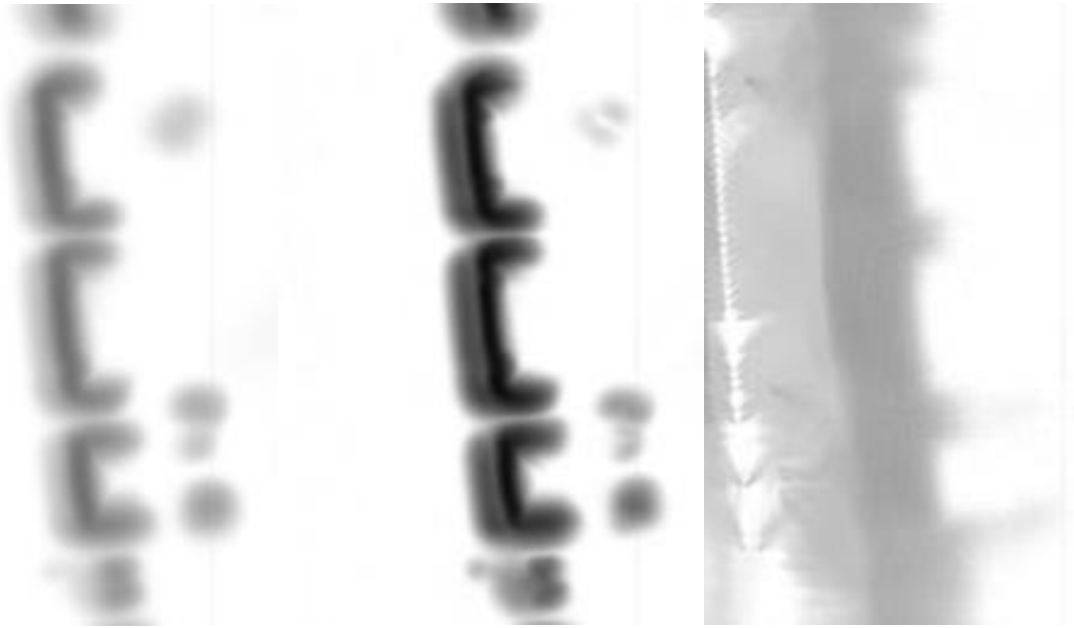
- Informative content required
- Integrate probabilities along line of sight from each position ς, respectively

$$P[Y_\varsigma] = P\left[\bigvee_i X_{\varsigma,i}\right] \qquad W_\varsigma = \int_\varsigma^\infty P[Y_{\varsigma^*}]d\varsigma^*$$



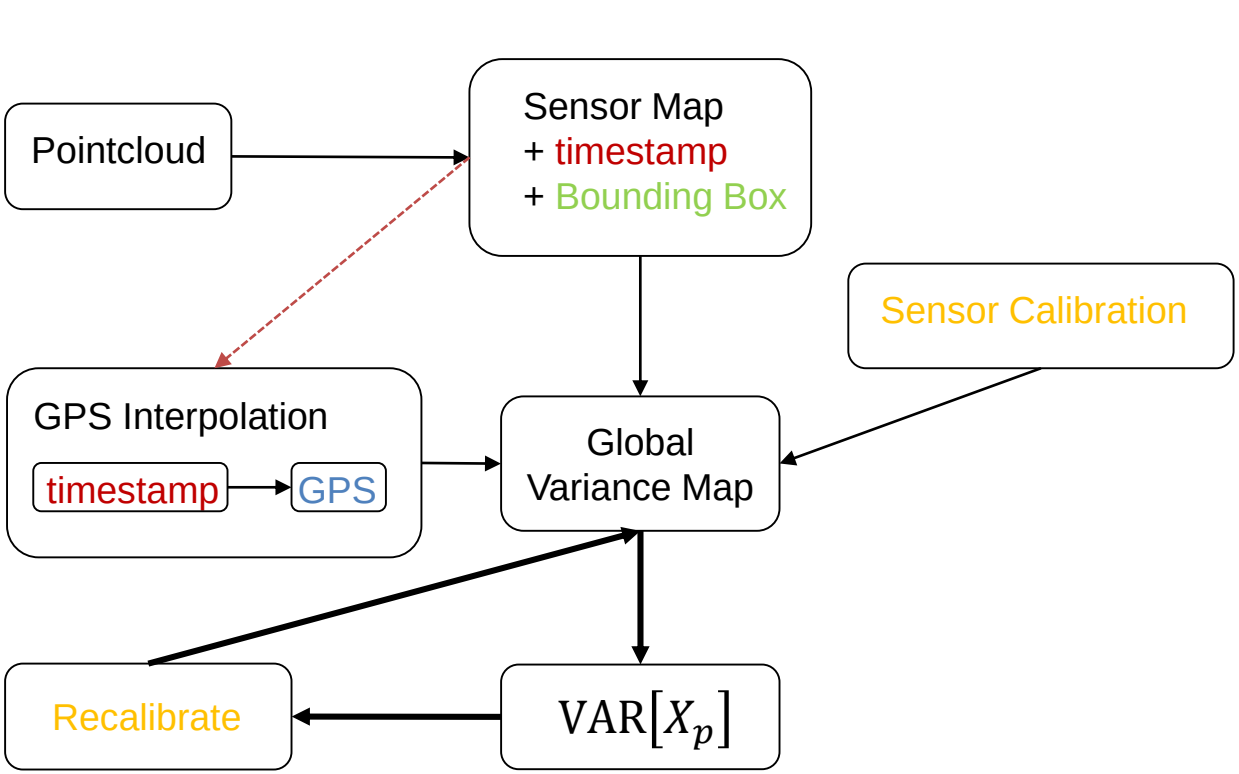
Global Variance Map

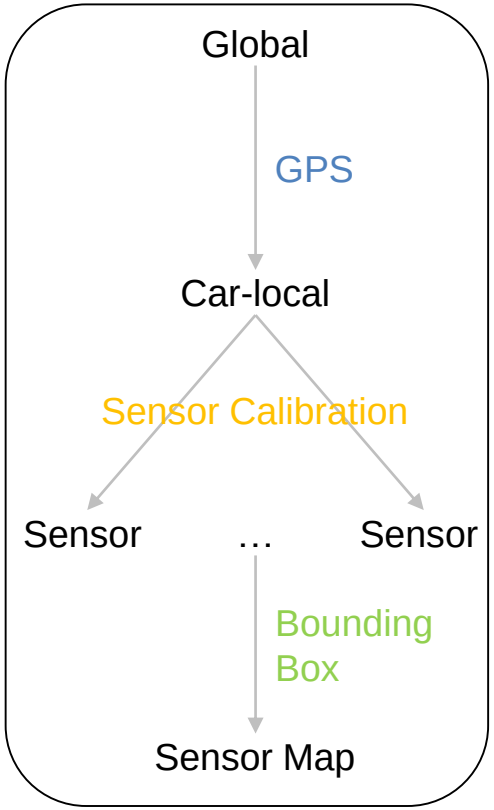
- Drive around and aggregate all sensor maps – Map-to-Map Matching
- Compute weighted mean and variance for each position
    - Rasterization with 2cm resolution
    - Left: Mean; Center: Variance; Right: Weight



Implementation Overview

- Simple implementation using vruin/CorONA scientific computation library
  - Easy computation on fields (e.g., computation of variance)
  - „Gradients are on the house“



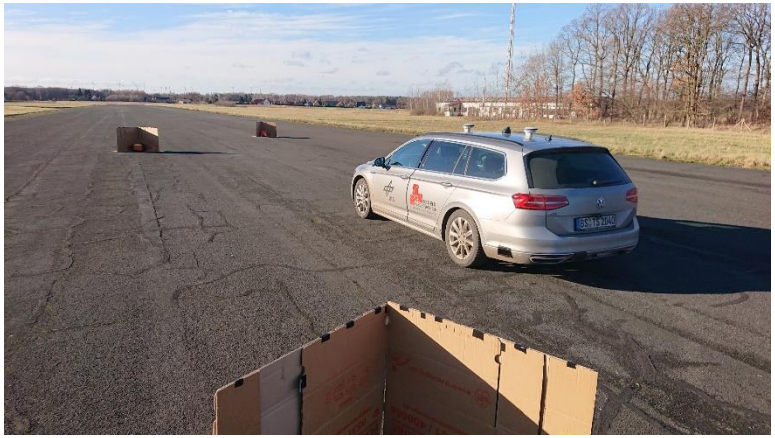


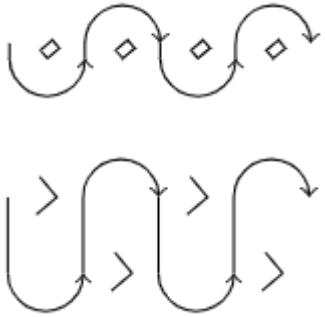
Outline

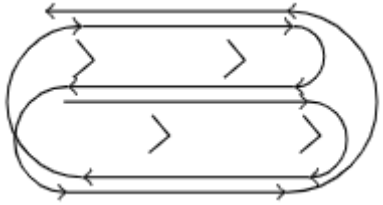
- Introduction
- Methodology
- Experimental Results
- Future Work

Testing Area

- Different testing areas
  - Edemissen testing area
  - DLR Braunschweig parking lot
  - Berlin roads
- Experiments regarding:
  - Different obstacles
  - Different layout
  - Different trajectories

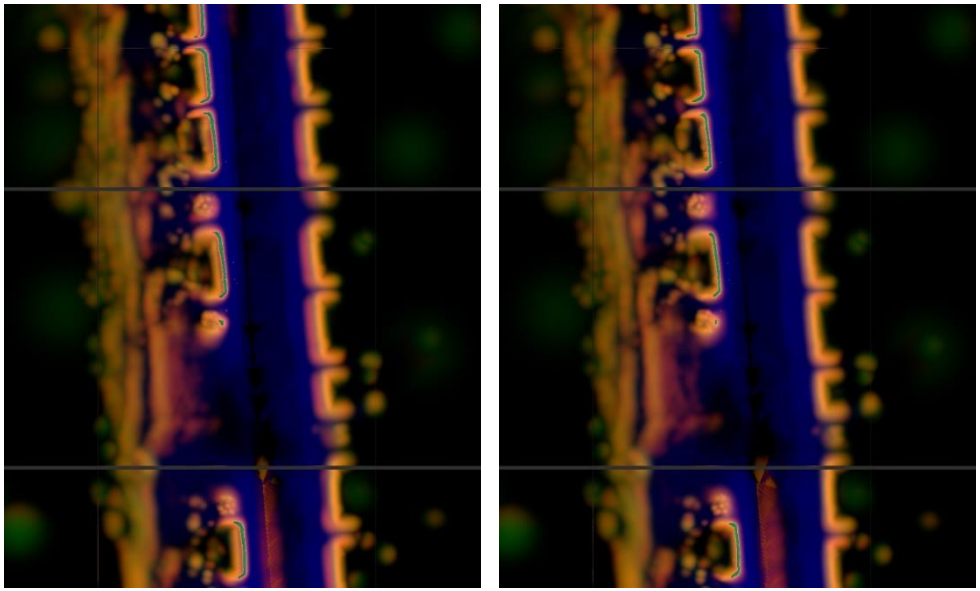


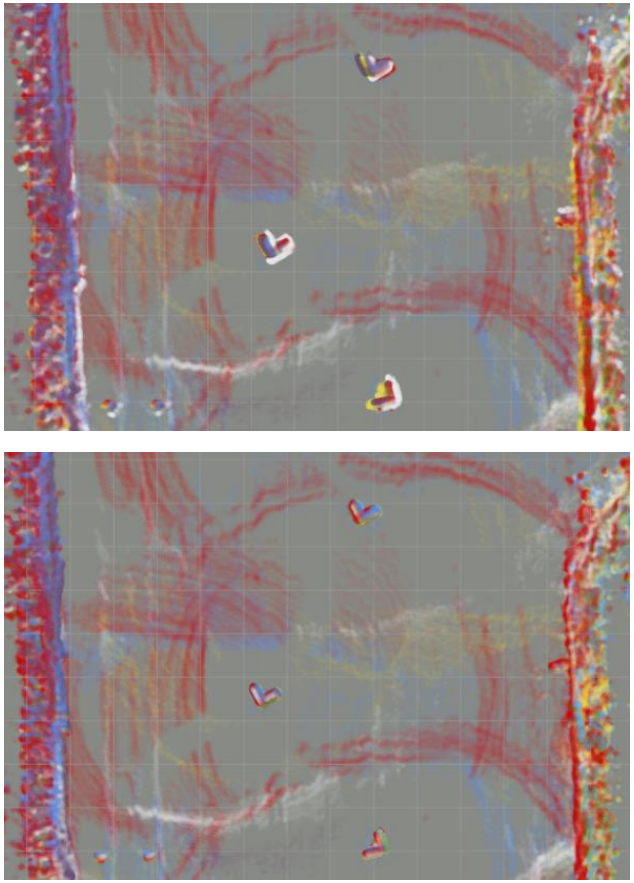




Visual Results

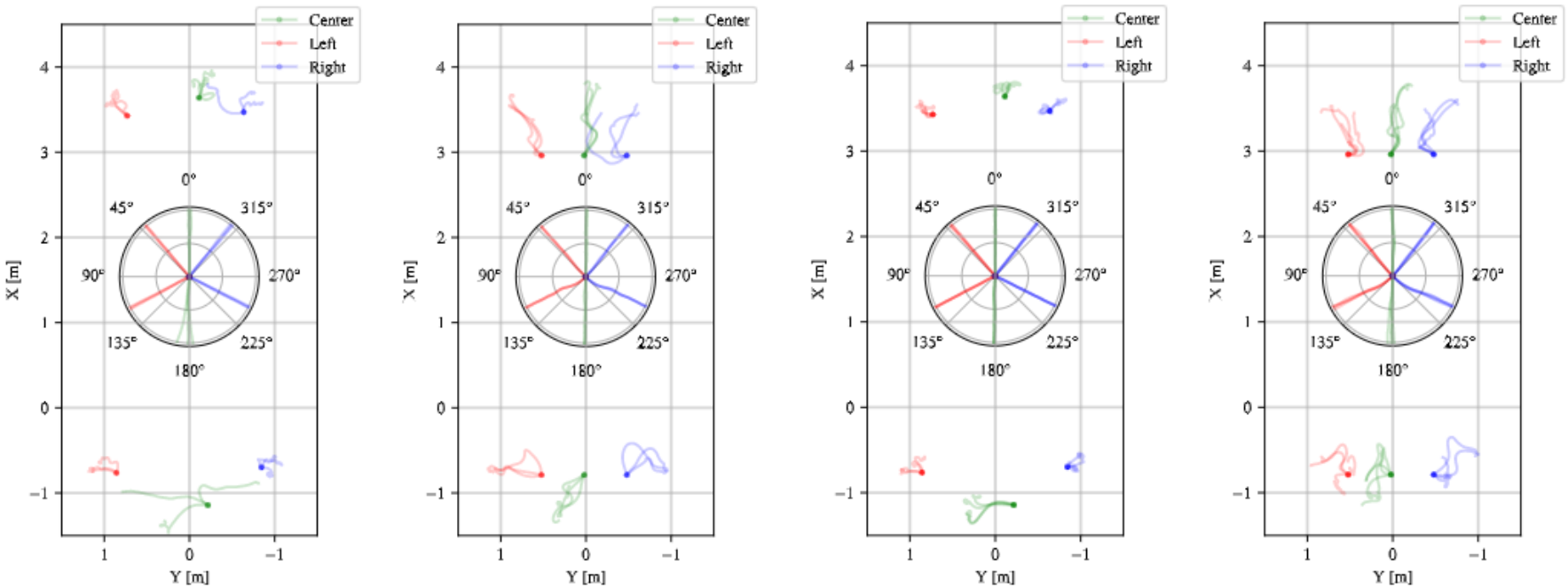
- Inconsistencies before and after calibration
- Left: on sensor maps
- Right: on point clouds





Results of Different Scenarios

- Examples for different Scenarios:
  - Accurate (image 1 and 3) vs. inaccurate (image 2 and 4) initial configuration
  - Boxes (image 1 and 2) vs. L-Boxes (image 3 and 4)
- Highly robust regarding angles; Fine-tuning regarding position required

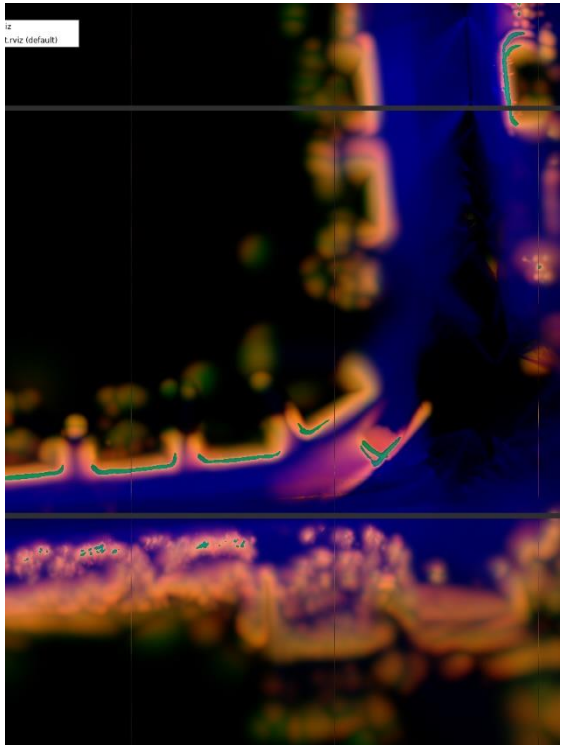


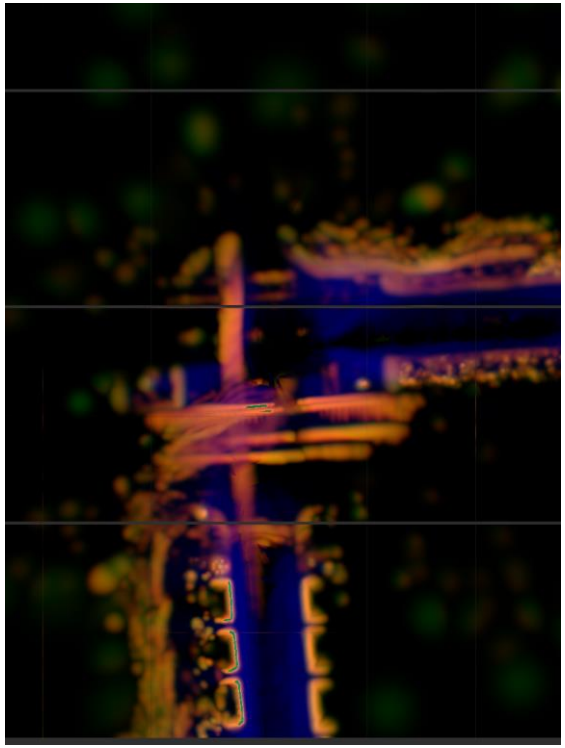
Outline

- Introduction
- Methodology
- Experimental Results
- Future Work

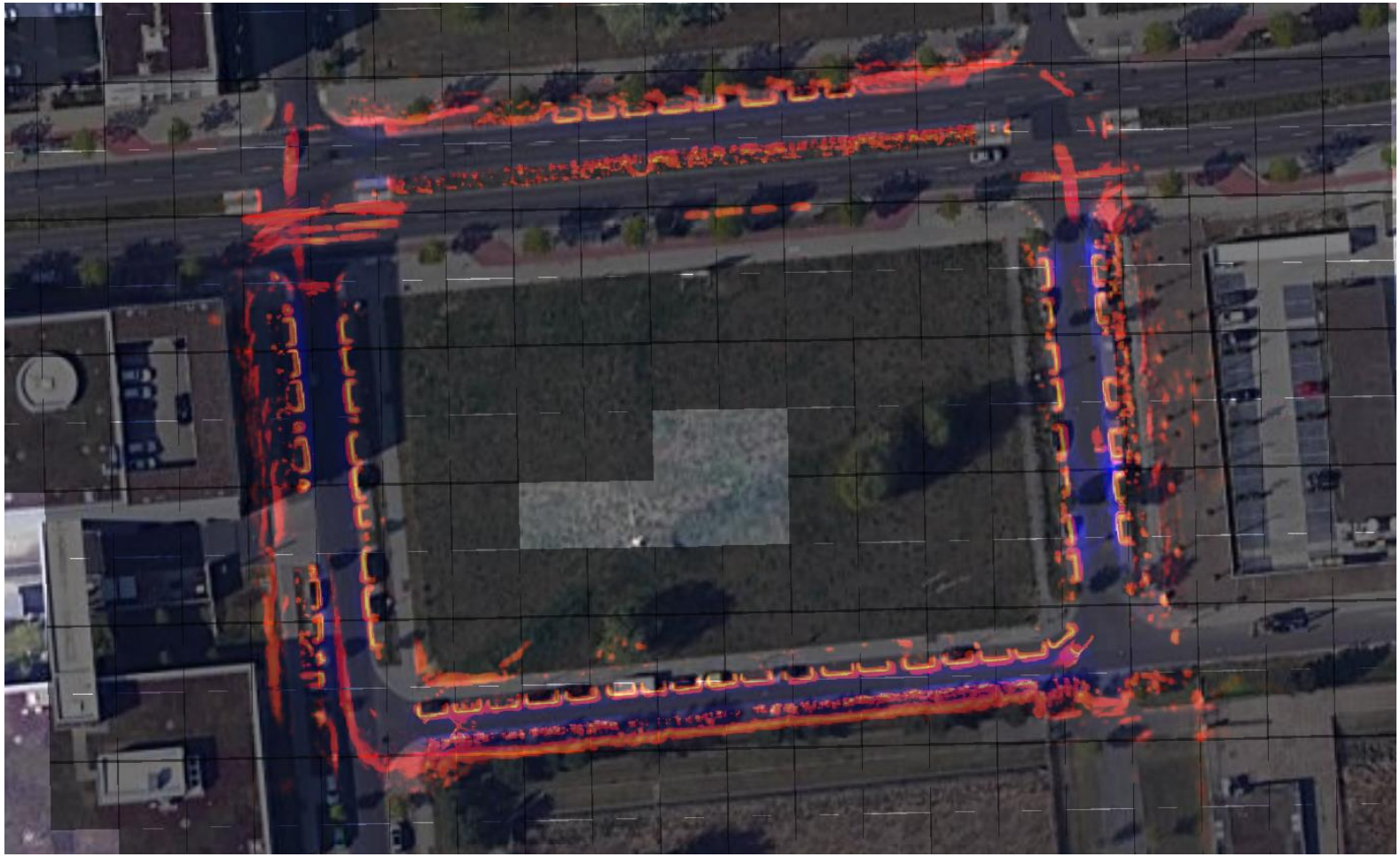
Future Work

- Artifacts: Dynamic Objects
  - Remove them via Semantic Segmentation Neural Nets
- Experiments on Camera based POIs
- Maximize Log-Likelihood
- Localization





Global Variance Map – Towards Localization



Satellite image from https://maps.google.de

Thank you for your attention!
Questions?