zenseact

VALIDATING SENSORS IN SELF-DRIVING CARS

Jonathan Ahlstedt /2021-11-01

Validating sensors in self-driving cars

- Company and team introduction
- System setup
- Object tracking
- Lane detection
- A typical day at work

zenseact

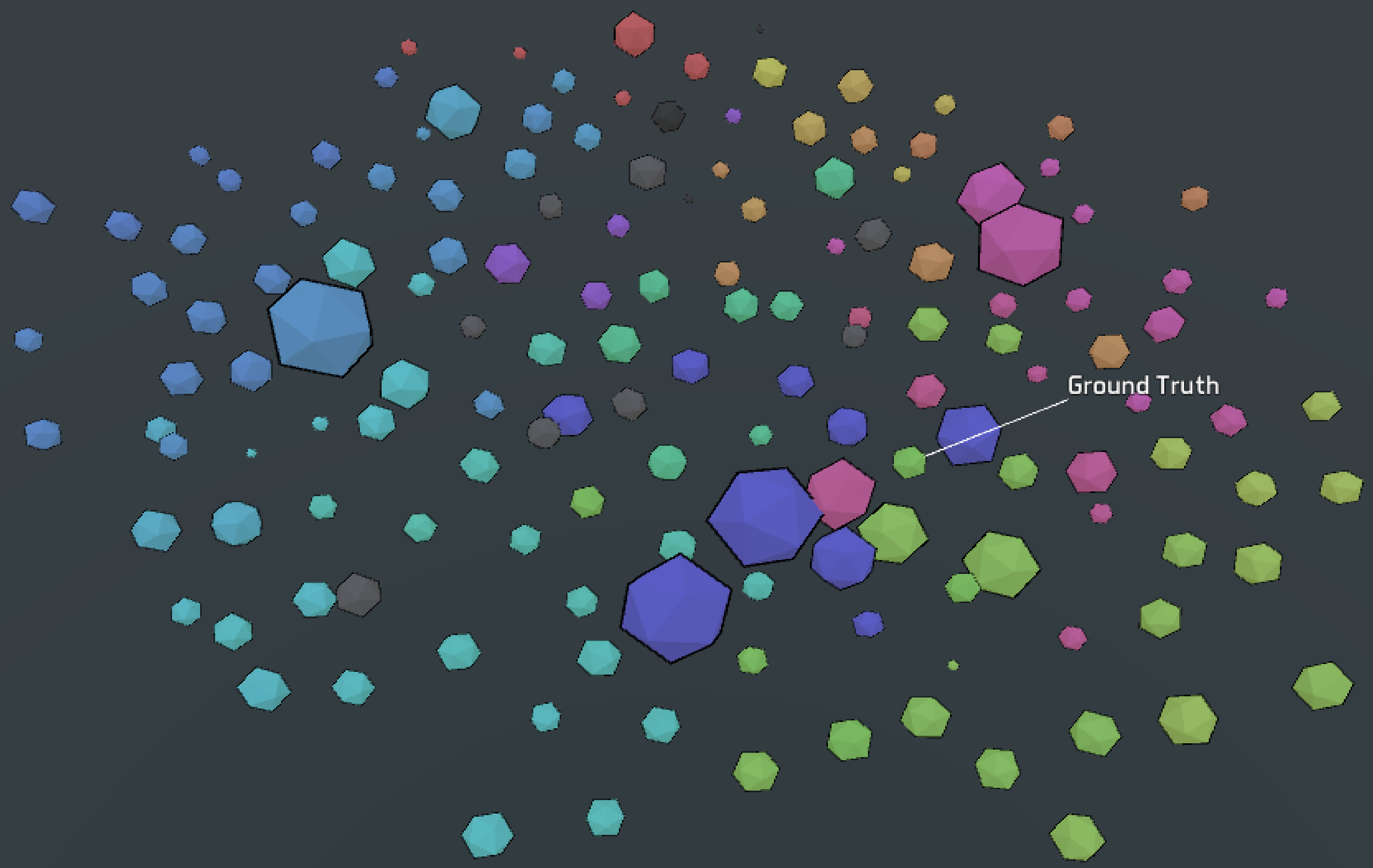
OUR BRAND PURPOSE

“To make safe and intelligent mobility real, for everyone, everywhere."

"Our purpose is our “reason for being” and marks our conviction to bringing ours, and our partners’ collective visions for autonomous mobility to life. And into the lives of everyone who desires it."

- Product Strategy & New Technology
- Collision Avoidance
- Connected Roadview
- Cruising and Highway Autonomous Driving
- IT
- Vision
- Shared Services

- Program/Product
- Strategy
- Business Enablement

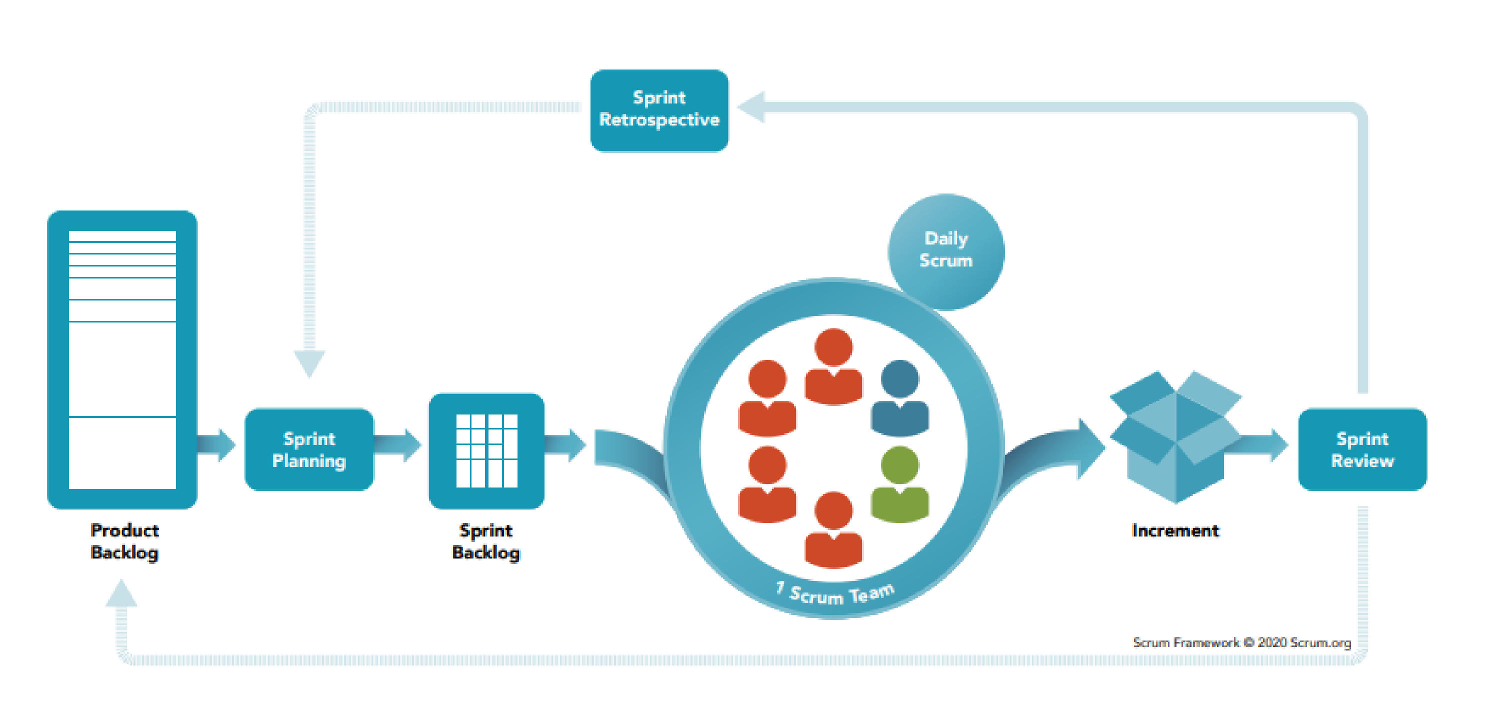


The Team

Product Owner

Engineering Manager

Agile development



Programing languages used

MATLAB python™ {C} PROGRAMMING BASH THE BOURNE-AGAIN SHELL docker

So we built an autonomous vehicle – What now?

How do we sell it?

- Legal aspects

Our product has to abides by current and future laws set by legislators around the world.

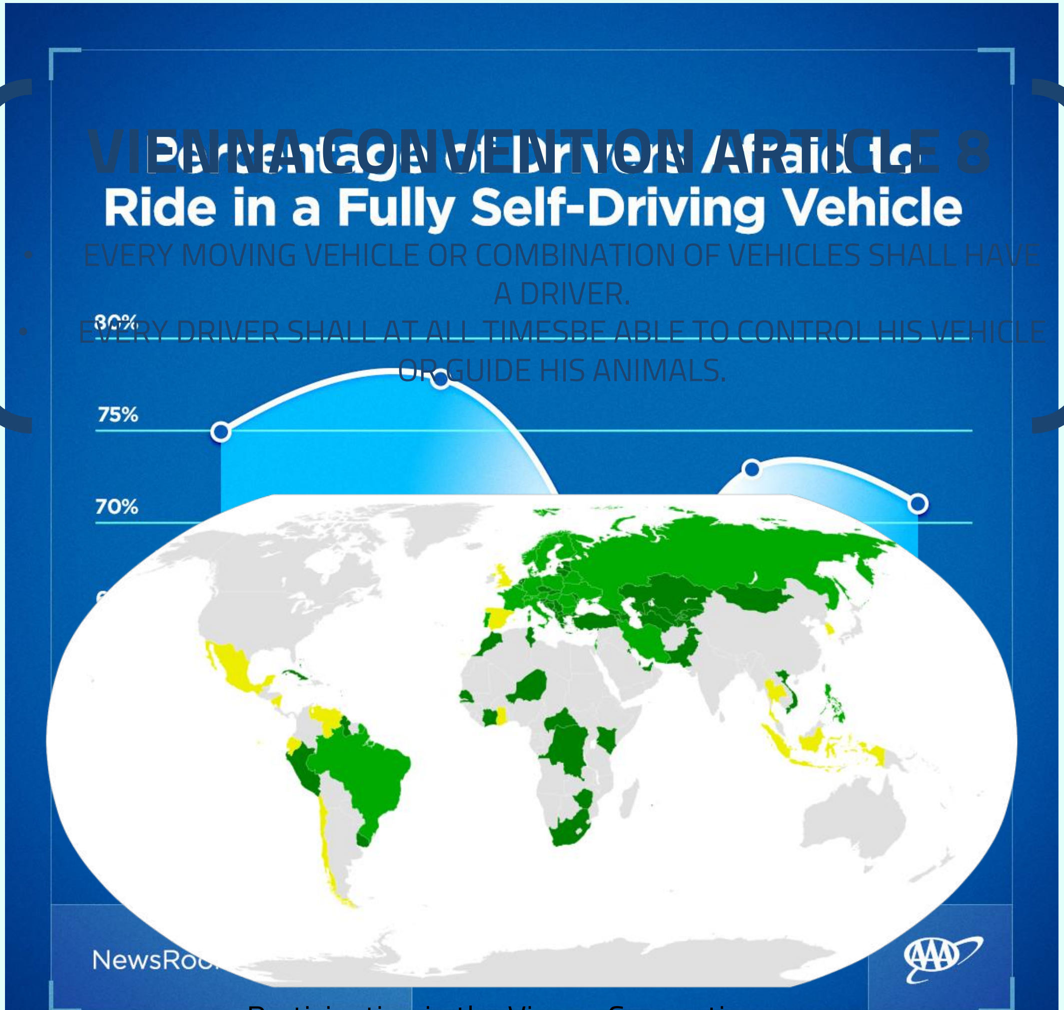
- Public opinion

How do we make people ride the cars when an error might be fatal?

We must prove to the world that our vehicles are safe!

VIENNA CONVENTION ARTICLE 8

- EVERY MOVING VEHICLE OR COMBINATION OF VEHICLES SHALL HAVE A DRIVER.
- EVERY DRIVER SHALL AT ALL TIMESBE ABLE TO CONTROL HIS VEHICLE OR GUIDE HIS ANIMALS.



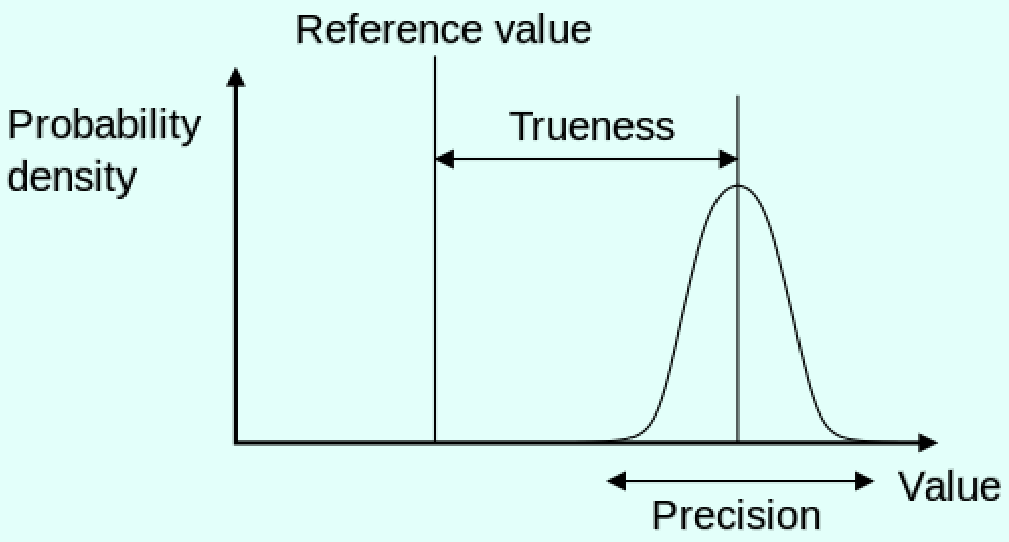
Participation in the Vienna Convention

Previous experience tells us that

- Functional Safety depends heavily on sensor performance
- Sensor specifications are usually not true
  - Suppliers sometimes overpromise
  - Suppliers sometimes do not know the statistical performance of their sensors
- Sensors that are to be used are sometimes not completed. Thus, no-one knows the true performance of the finished product

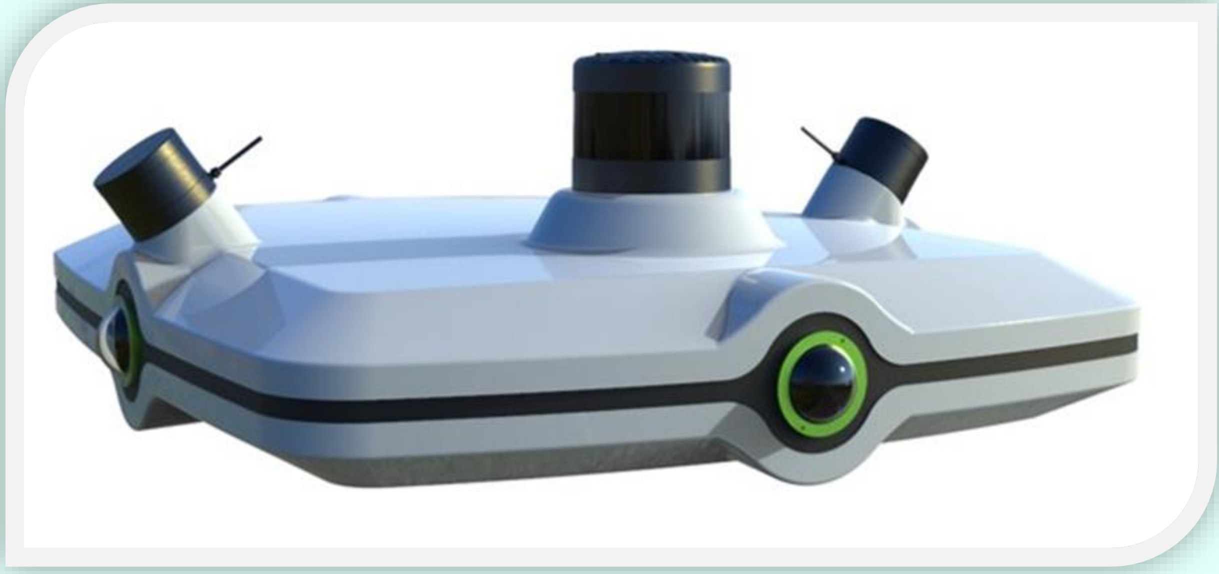
- So how do you measure sensor accuracy?

Simple!
Compare sensor output to a reference



Our solution – Ground Truth Roofbox

Video introduction



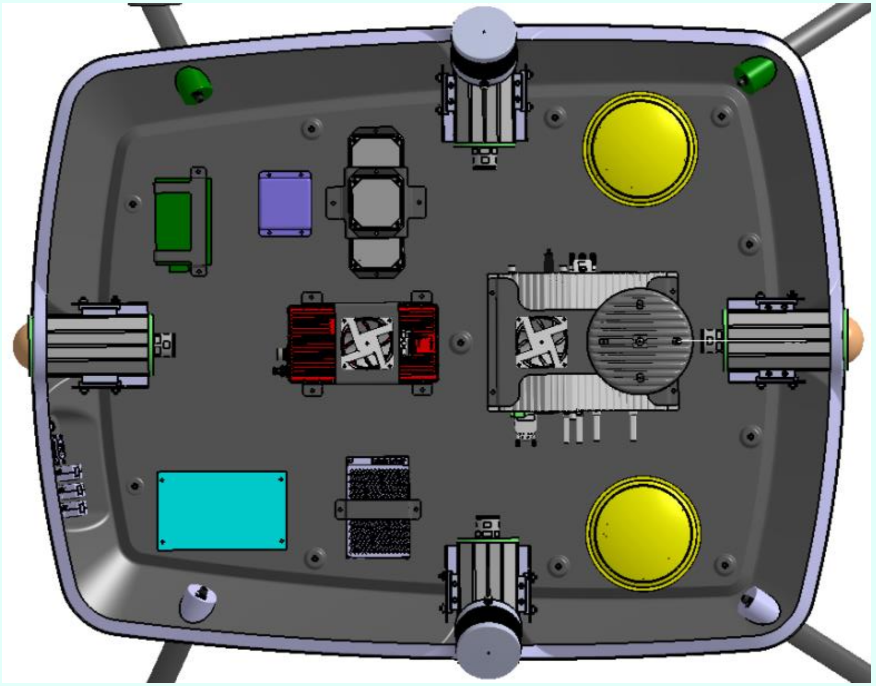
Use cases and features

- Reference sensor data in real traffic is needed to:
  - Reduce manual verification
  - Extend test-track verification to real-world
  - Automate verification
  - Perform statistical verification

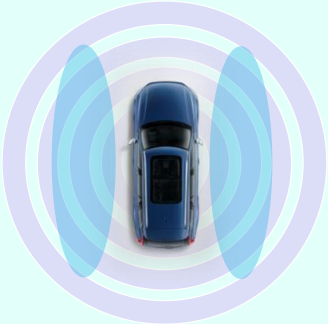
- The system provides 360 degree reference data of the environment of the vehicle and is easy to use

- Collected data is automatically post processed offline, documented and formatted for use in verification and development

- Inhouse-developed Roofbox and Software includes
  - Lidar (range ~200 m)
  - Cameras
  - GPS/IMU



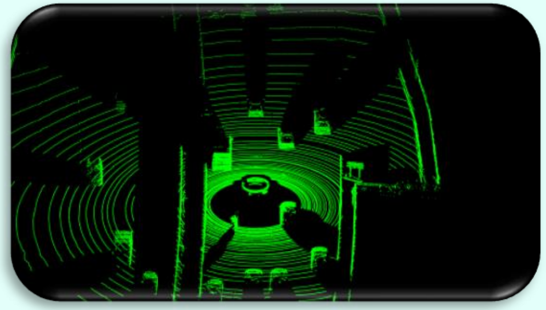
Camera FOV



Lidar FOV

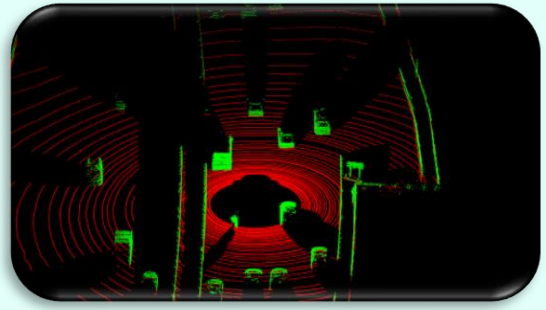
Object tracking – step by step

Raw pointcloud

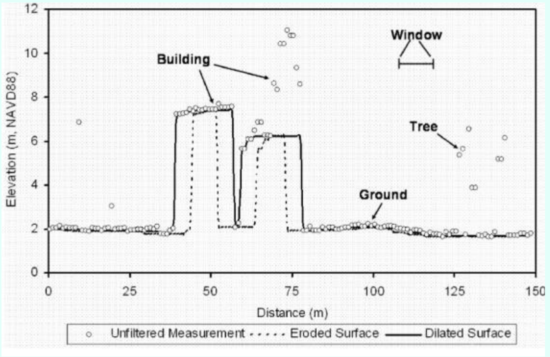


- 128 + 16 + 16 Laser layers
- 9Hz rotations
- ~260000 points per rotation
- X, Y, Z, Intensity

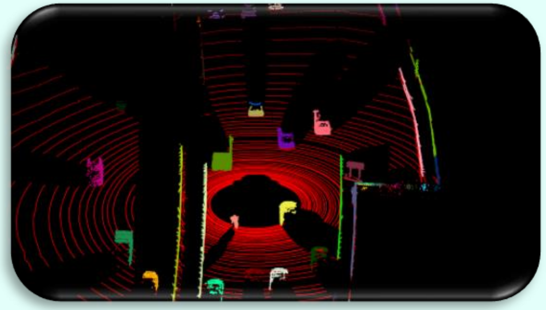
Ground removal



- Adaptive progressive morphological filtering



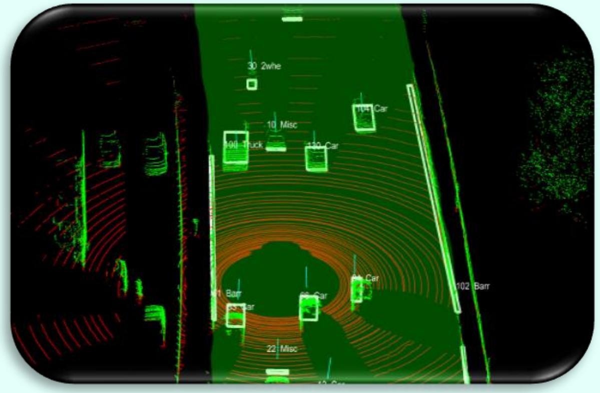
Object segmentation



- Spherical conditional Euclidian clustering

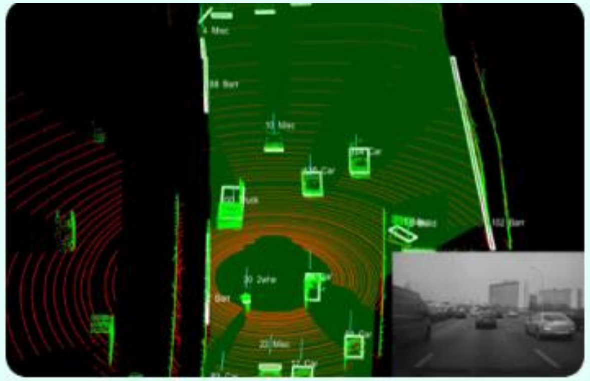
Object tracking – step by step

Initial classification



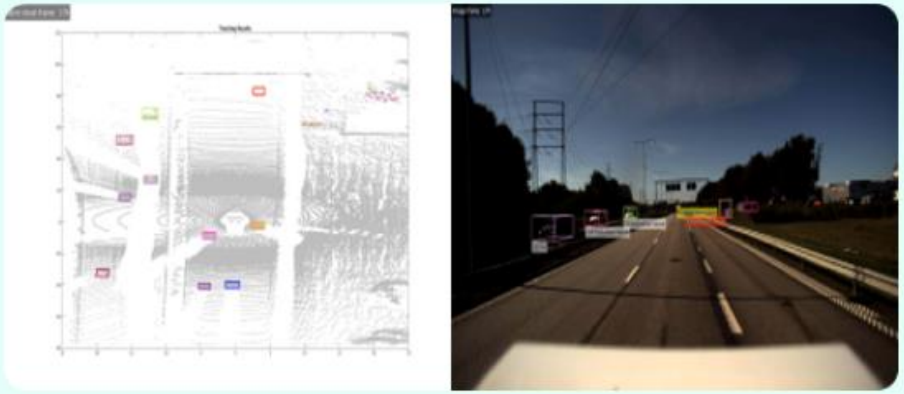
- Support vector machine
    - LibSVM
    - Single-frame
    - Geometry-based

Tracking and post-classification



- Independent Forward and backward pass Kalman filter
- Merge tracklets
- Rauch-Tung-Striebel smoothing

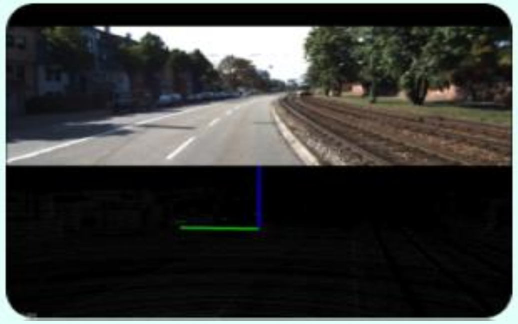
Camera fusion



- Merge with camera tracking and classification

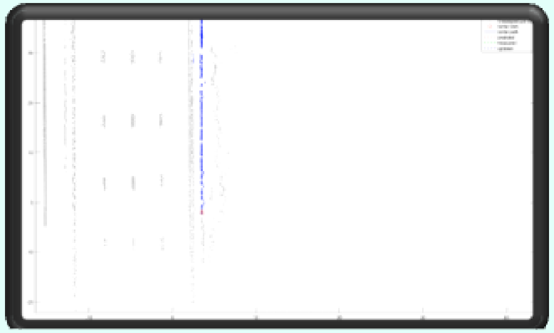
Lane detection – step by step

Lane marking detection



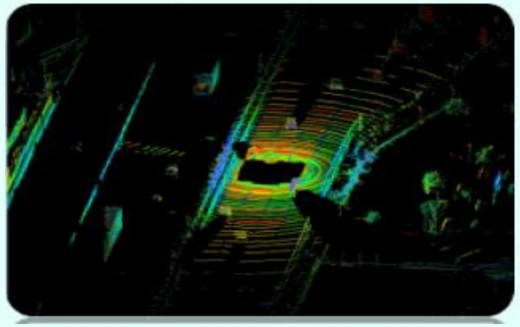
- Find lane marking points by looking at intensity

Lane marker tracking



- Track the center of lane markings to filter out connected lane marking segments

Lane detection



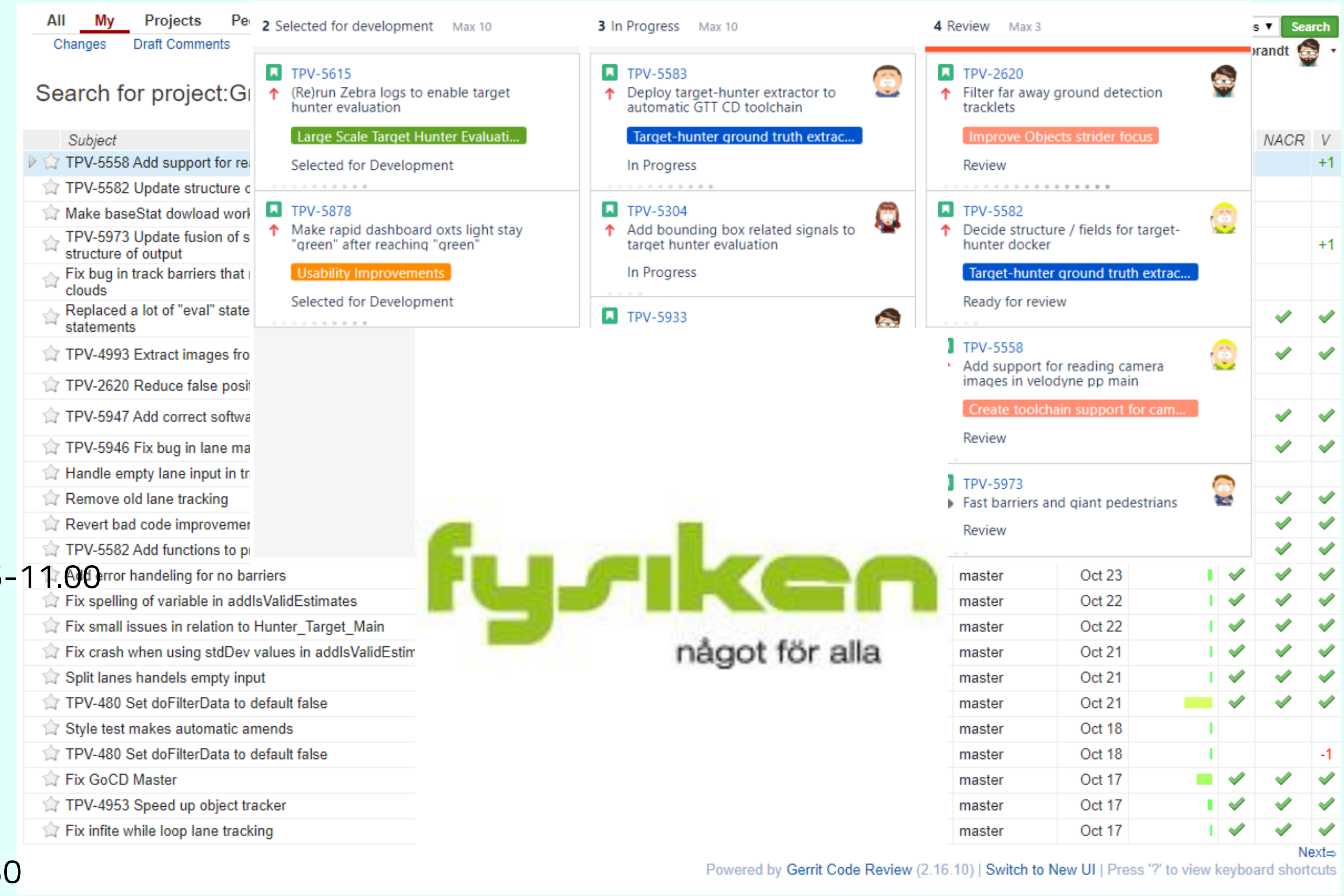
- Use tracked lane centers to find where lanes are

Camera fusion

- Find lanes using high reflectivity points in camera

A typical day at work

- Get to office at 8.00
  - Review others code
  - Answer questions in slack (Chat app)
  - Work on small coding patch
  - Check in with other teams
- Standup at 10.00
  - Sprint Goal
  - What did you do?
  - What will you do?
  - Is anything blocking you?
- 80% of team does lunch training 10.15-11.00
  - Gym (Monday/Wednesday)
  - Running (Tuesday/Thursday)
- Lunch 11-12
- Work 12-14
  - Develop new feature
  - Attend a meeting with another team
- Team gets some air (and fika) 14-14.30
- 14.30-17~ Continue coding until I leave



zenseact

Make it real.

Bonus slides

Same team, three different companies

~120 employees

VOLVO

~60 employees + Money

veoneer

~700 employees

ZENUITY
Make it real.

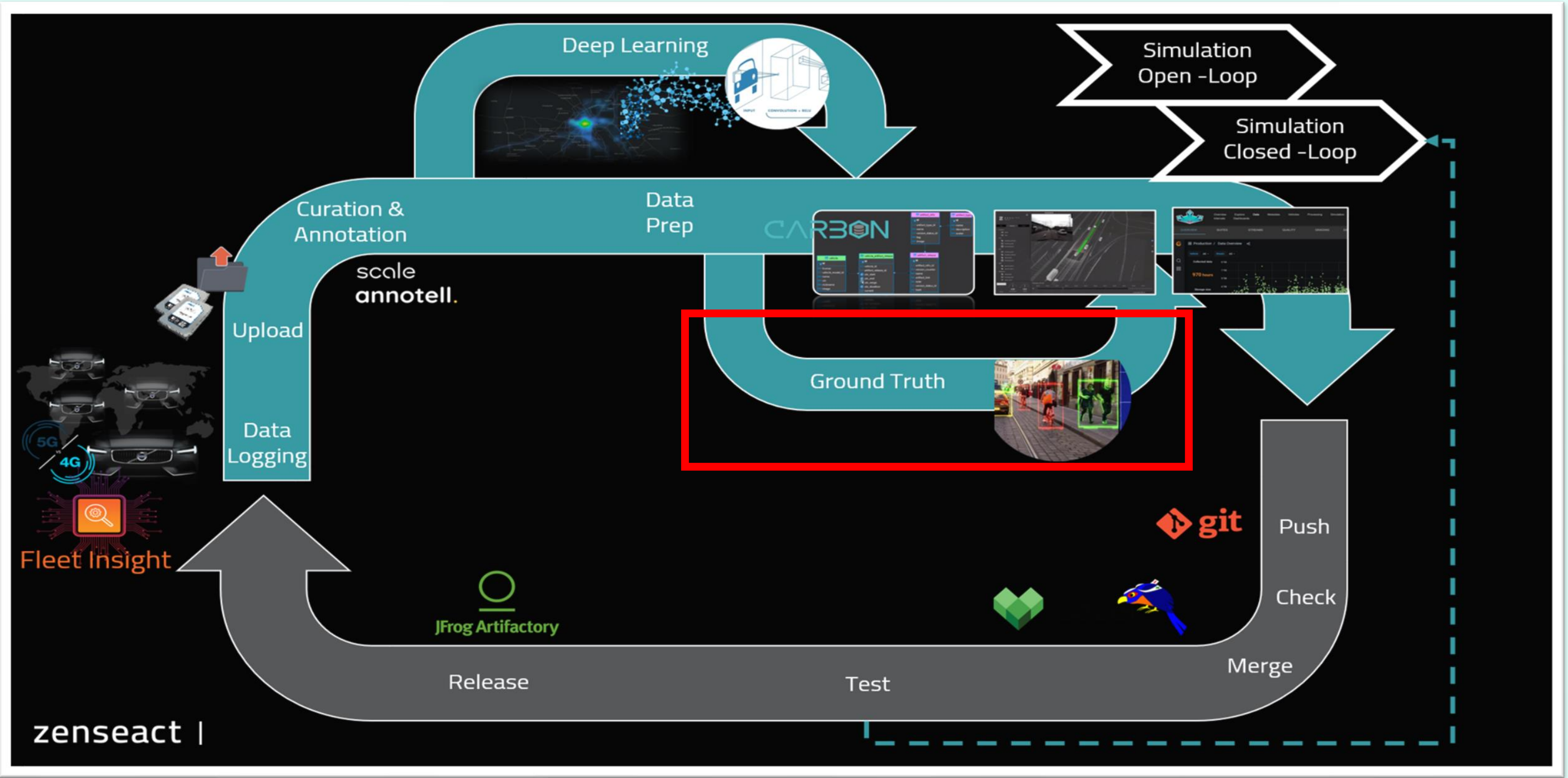
Sweden office

~500 employees

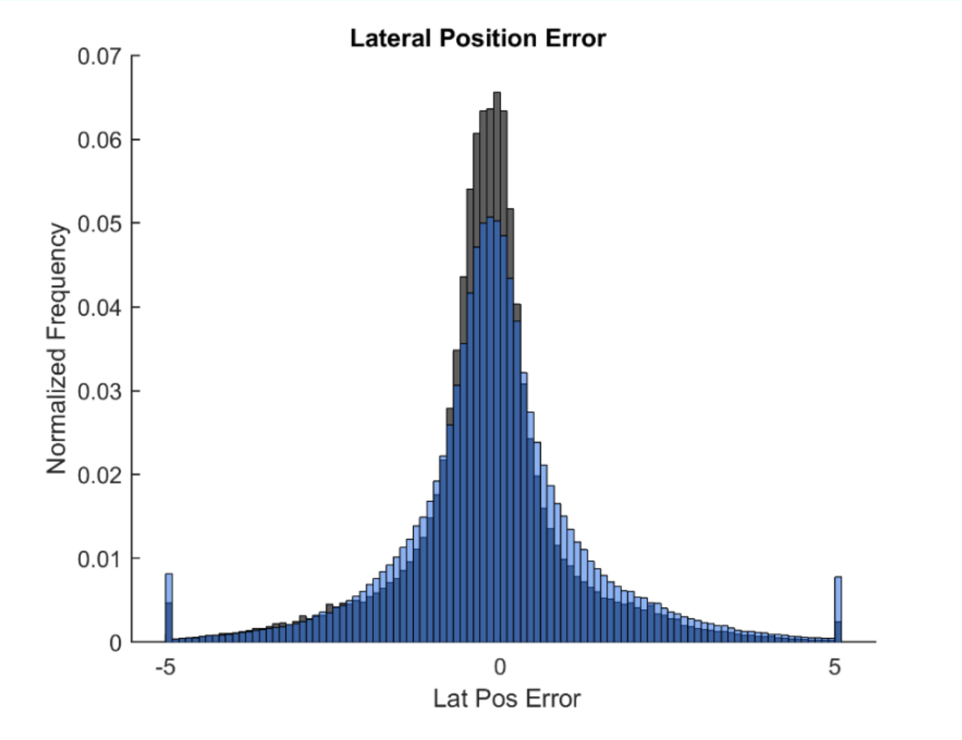
zenseact
Make it real.

Us + Germany office

veoneer


Deep Learning
Simulation
Open -Loop
Simulation
Closed -Loop
Curation &
Annotation
Data
Prep
CARBON
scale
annotell.
Upload
Ground Truth
Data
Logging
5G
4G
Fleet Insight
git
Push
Check
JFrog Artifactory
Merge
Release
Test
zenseact

Typical sensor evaluation with GT



Ghosts

To properly label production sensor objects as ghost, there has to be extremely high availability of the reference when comparing with ground truth (the physical environment). The definition below is adapted to be applied using Zenuitys roofbox system as reference, highly relying on Velodyne lidar(s) and associated software, a sensor setup with poor long range performance. This is why the labeling of ghost objects works at reference detection level with high availability together with object level with tracks. It is also the reason for why there is a range threshold, L_{limit}. A production sensor object $\mathcal{T}^{s,j}$ is labeled as ghost if $\mathcal{T}^{s,j}$ is not matched to any reference object $\mathcal{T}^{r,i}$ using the matching algorithm and $\mathcal{G}_k^j$ is true for at least 1/3 of all k during the lifetime of $\mathcal{T}^{s,j}$ and while the distance between $\mathcal{T}_k^{s,j}$ and the host vehicle is less than L_{limit}.

$$\mathcal{G}_k^j = d_{k,det}^{s,j} > r_{thld}^{det} \vee \left[d_{k,obj}^{s,j} < r_{thld}^{obj} \wedge \left[(|dV_{i,j,k}^{lat}| > v_{thld}^{lat}) \vee (|dV_{i,j,k}^{lgt}| > v_{thld}^{lgt}) \right] \right]$$

$d_{k,det}^{s,j}$: Distance between $\mathcal{T}_k^{s,j}$ to the closest reference detection (bounding box) at time frame.

$d_{k,obj}^{s,j}$: Distance between $\mathcal{T}_k^{s,j}$ to the closest reference object track (bounding box) at time frame.

$dV_{i,j,k}$: Velocity difference between $\mathcal{T}_k^{s,j}$ and reference object corresponding to $d_{k,obj}^{s,j}$.

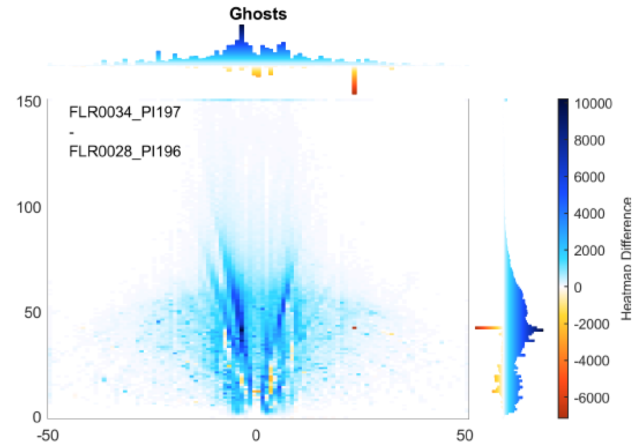
$r_{thld}^{det} = 3.0$: Range threshold.

$r_{thld}^{obj} = 3.0$: Range threshold.

$v_{thld}^{lat} = 2.5$: Velocity threshold.

$v_{thld}^{lgt} = 2.5$: Velocity threshold.

$L_{limit} = 60$: Threshold for distance between $\mathcal{T}_k^{s,j}$ and host vehicle.



Global/Local positioning GT

- OXTS RT3003
- GPS and IMU data is fused with base station information
- Result:
  1cm globally accurate position