



# AI in Future Mobility

## Introduction & Sensors

Prof. Dr. rer. nat.  
**Toralf Trautmann**  
Faculty Mechanical Engineering

T +49 351 462 2383  
[trautmann@htw-dresden.de](mailto:trautmann@htw-dresden.de)

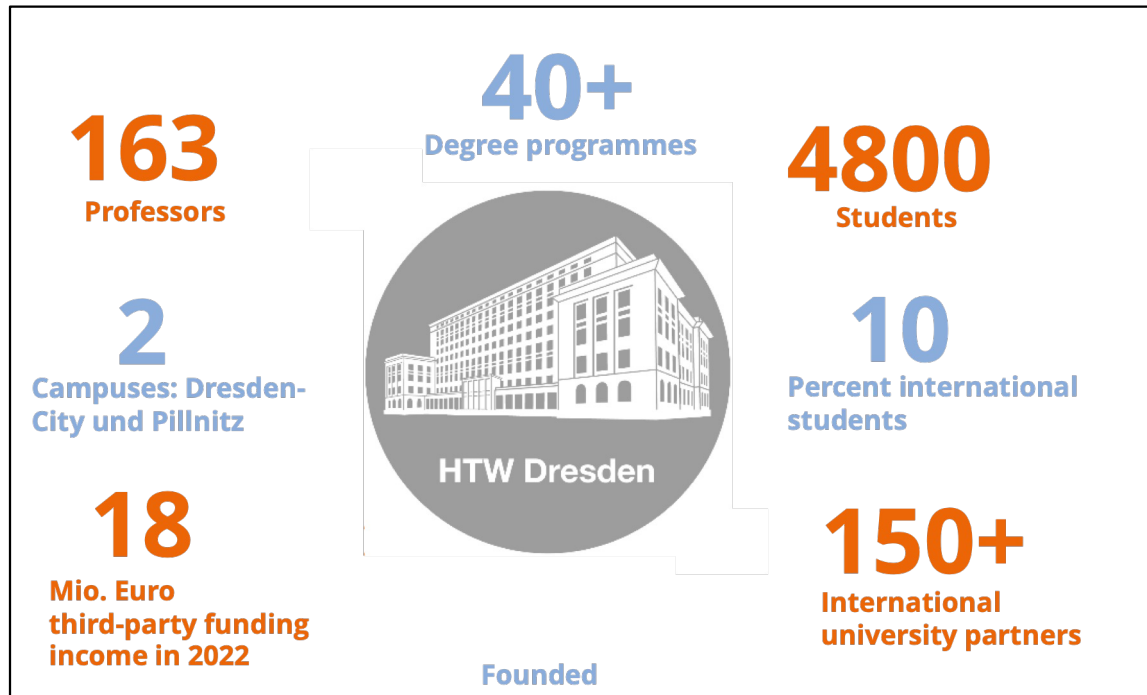




# The HTWD and the Mechlab Laboratory







### Prof. Dr. rer. nat. Toralf Trautmann (Physicist)

1967: born in Zella-Mehlis (Free State of Thuringia),  
1995 - 2000: PhD research at Freiberg University of Technology,  
Thesis: „Radioluminescence of Feldspar“  
2001 - 2005: Development Engineer  
Robert-Bosch-GmbH, Schwieberdingen  
since 2005: Professor of Automotive Mechatronics (HTWD)



### Research/Education:

- Development and test of advanced driver assistance systems,
- Surrounding sensors (**Lidar**, Radar, Camera) and data fusion,
- **Type approval process and PTI for autonomous vehicles**,
- Software development and prototyping using Matlab/Simulink/ROS.

### Stuff:

- 1 lecture assistant and 2 full-time research assistants
- 3 student workers (automotive engineering)
- 1 guest student (University of Primorska, Slovenia)





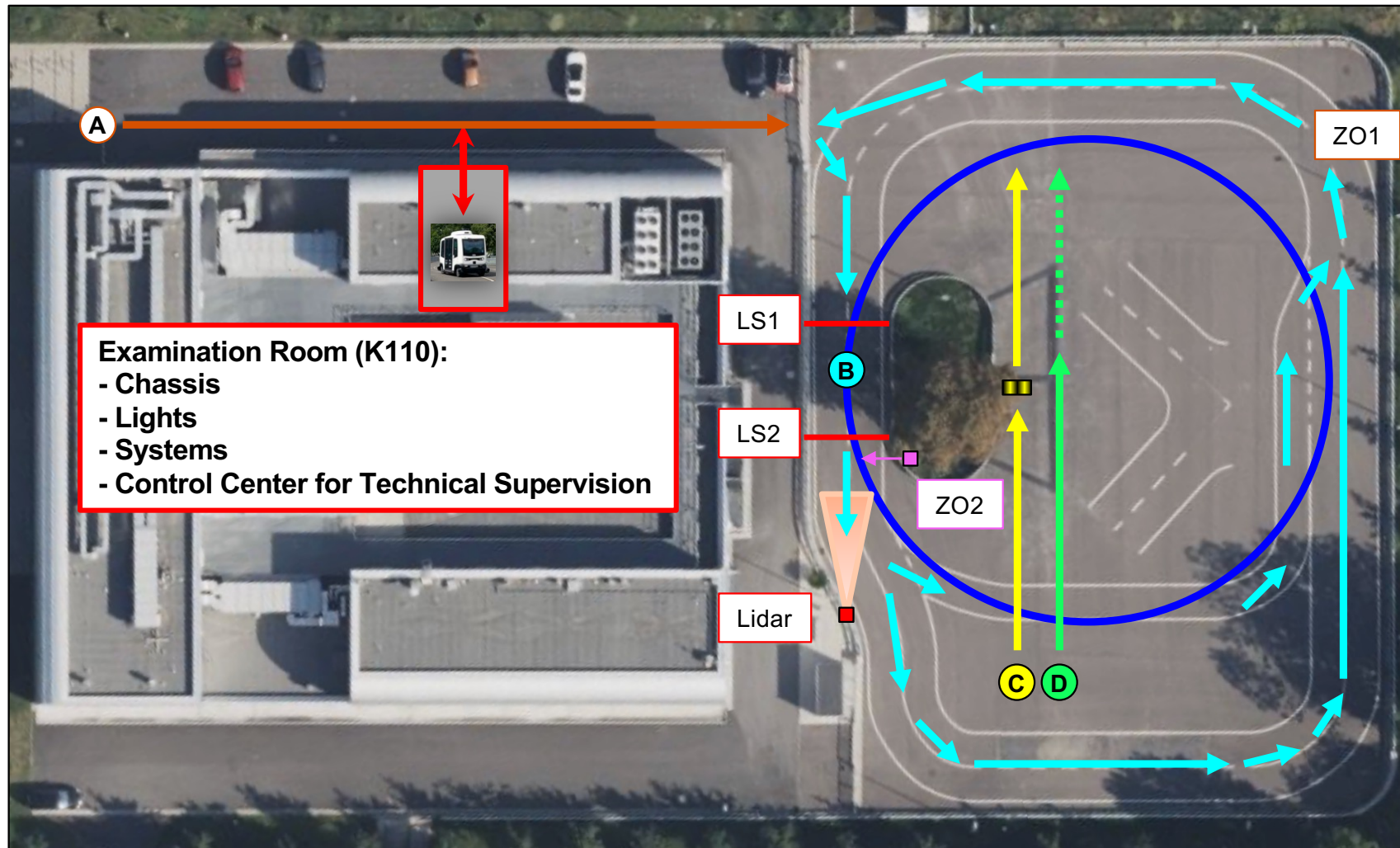
### Partners:

- tracetrone GmbH
- FSD Fahrzeugsystemdaten GmbH
- GTÜ GmbH
- Accident Research Institute
- Bertrand GmbH
- Dewesoft d.o.o. (Slovenia)

### Universities:

- TU Dresden
- Univ. of Primorska (Slovenia)
- Univ. of the Balearic Islands (Spain)

- Different test cars (Skoda Enyaq, MB EQE, Passat GTE, Fiat Panda)
- **Autonomous Shuttle (EasyMile EZ10)**
- Matlab with all toolboxes
- Lidars: Velodyne, Ouster, Livox, LS-Lidar (1550nm), OPSYS
- Radar: Continental SRR & LRR











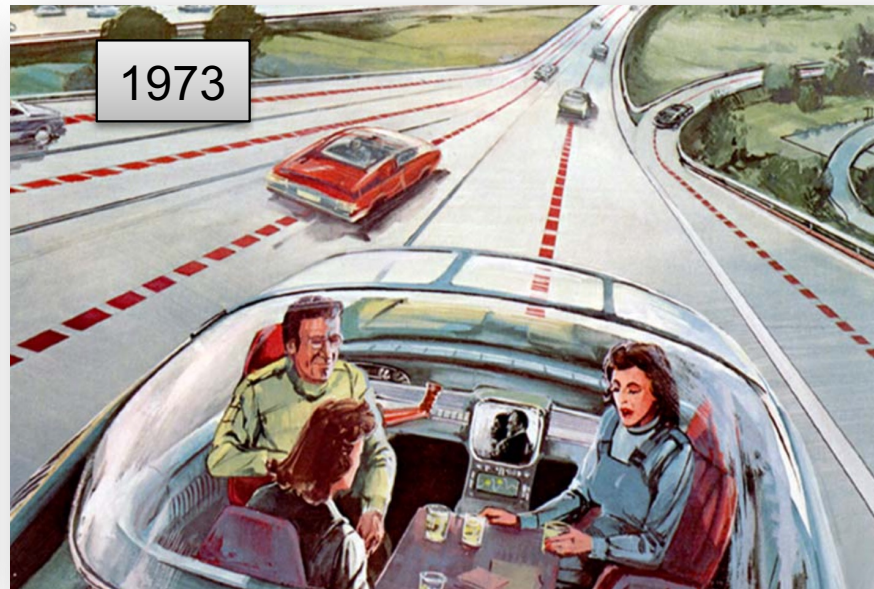
# Brief history and status of Driver Assistance Systemes and Autonomous Driving





## Short History

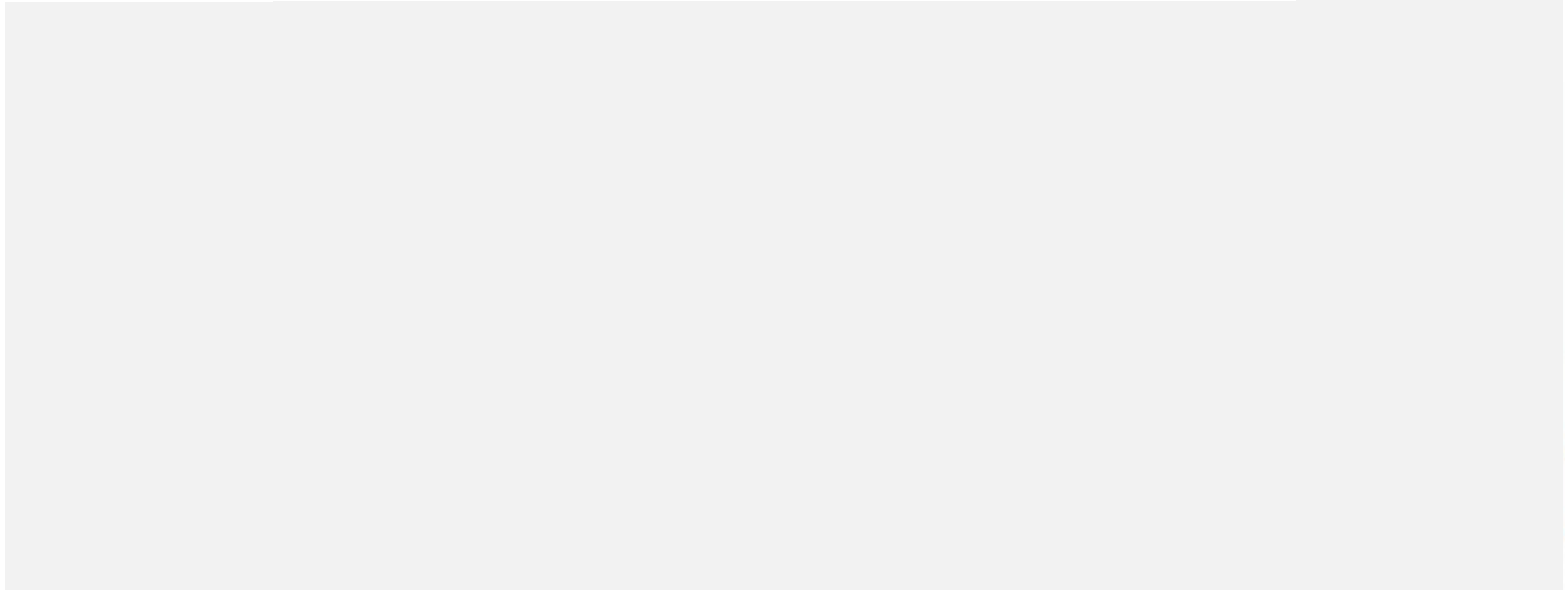
- 1925: Remote-controlled vehicle in normal rush-hour traffic in New York
- 1958: GM FireBird III Concept Car
- Joystick instead of accelerator, brake pedal, and steering wheel
  - "Autopilot" via induction loops
- 1985: Start of VaMoRs program of the Bundeswehr University (Germany)
- 1987 First autonomous high-speed test drive (20 km)
  - 1995 First Long-distance test drive (1750 km, > 95% autonomous)



## Human

## Human & Car

## Car

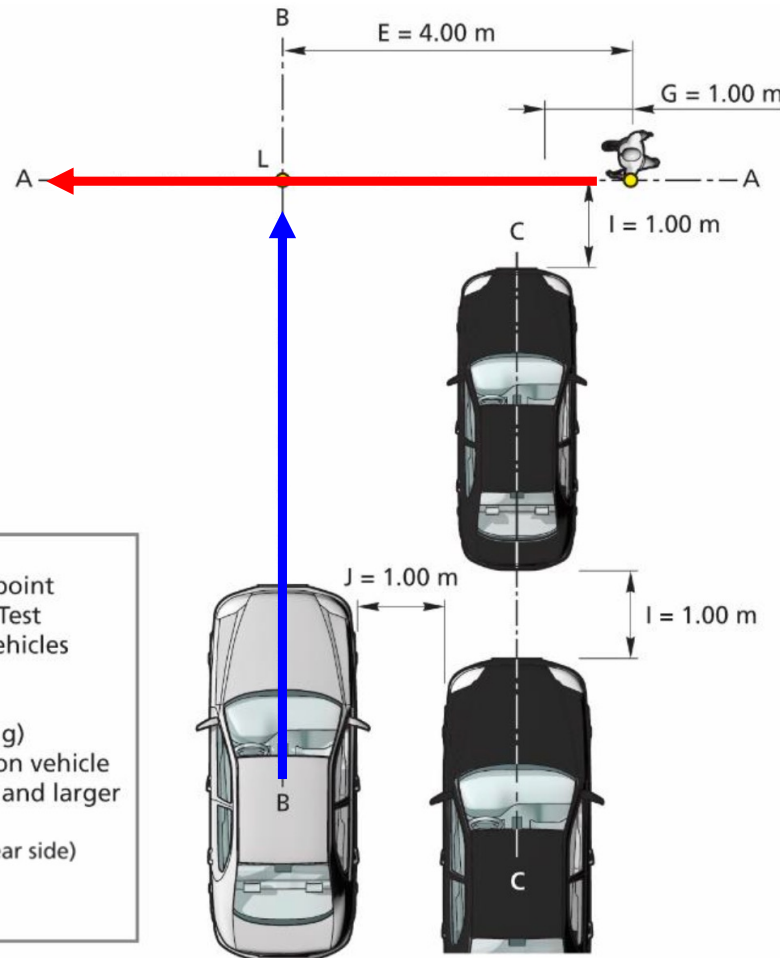




## Examples of current driving functions (Level 2+)



2024: EURO-NCAP → The test procedure will be slightly adjusted.



## Axes

AA – Trajectory of pedestrian dummy H-point  
BB – Axis of centreline of Vehicle under Test  
CC – Axis of centrelines of obstruction vehicles

## Distances

G – Dummy acceleration distance (running)  
I – Dummy H-point to front of obstruction vehicle  
J – Distance between Vehicle under Test and larger obstruction vehicle  
E – Dummy H-point, start to 50%-impact (near side)

## Points

L – Impact position for 50% scenarios

Test dummies  
keep moving  
constantly and  
do NOT stop!

Not necessarily  
realistic!

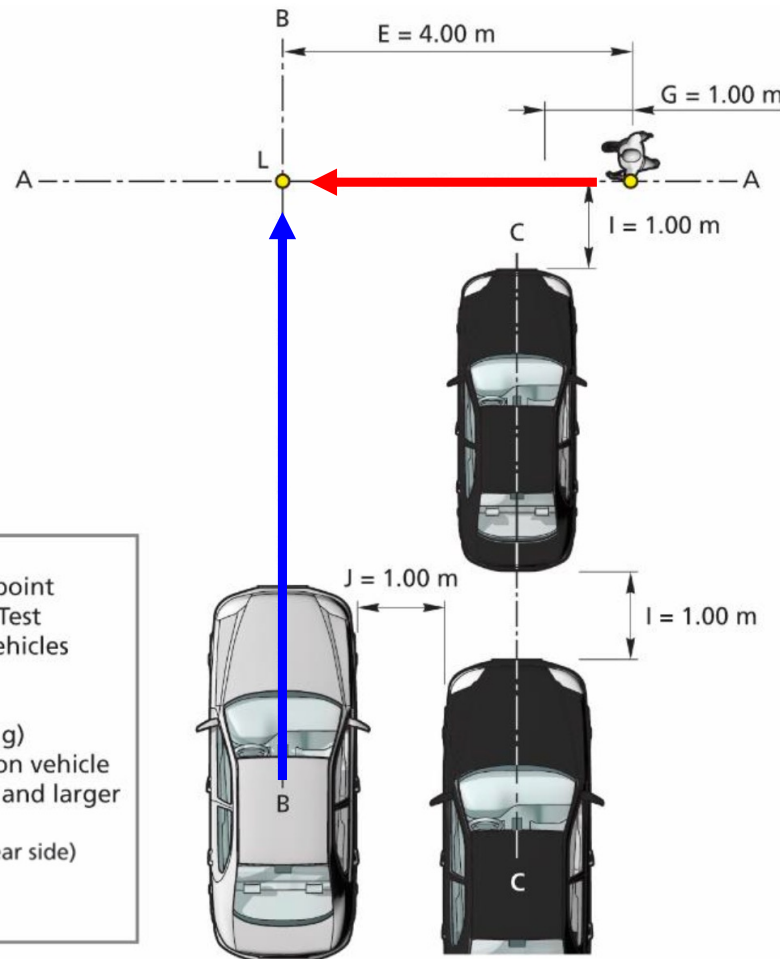
**Figure 7-3: CPNCO-50 scenario, Running Child from Nearside from Obstruction (Annex A)**





**AUTO  
MOBIL**

**VW Nachtest**



Adjustment: Child dummy remains stationary (at approximately 25% overlap)

**Figure 7-3: CPNCO-50 scenario, Running Child from Nearside from Obstruction (Annex A)**





**AUTO  
MOBIL**



**AUTO  
MOBIL**







**AUTO  
MOBIL**

***Nio 1. Versuch***

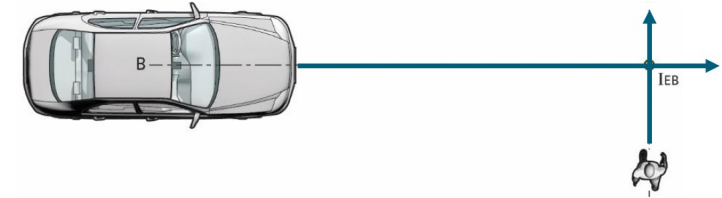


# Requirements for Type Approval

Car2Car



Car2Ped



| Nr. | Aim                                | $v_{VUD}$ / km/h | unb. | bel. |
|-----|------------------------------------|------------------|------|------|
| 1   | Car (stationary)                   | 20               |      |      |
| 2   |                                    | 42               |      |      |
| 3   |                                    | 60               |      |      |
| 4   | Car<br>(moving with 20 km/h)       | 30               |      |      |
| 5   |                                    | 60               |      |      |
| 6   | Pedestrian<br>(moving with 5 km/h) | 20               |      |      |
| 7   |                                    | 30               |      |      |
| 8   |                                    | 60               |      |      |

max. 10% error of  
individual tests

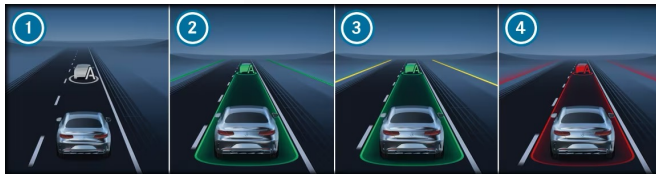
max. 10% error of  
individual tests

2 successful attempts out of a maximum of 3 attempts

# Example of highly automated driving (Level 3)

## Driving Assistance Package with DRIVE PILOT

Parking Package with 360°-Camera



Front Long-Range Radar  
Opening angle 90° / 9°

Introduced at the end of  
2023, only in the S-Class  
(approx. 6.500,- €)

Discontinued at the  
beginning of 2026

LiDAR  
Opening angle 120°

Multi-Mode Radar  
4x, Opening angle 130°

Stereo Multi-Purpose Camera  
Opening angle 70°

Antenna Module

Rear Multi-Purpose Camera  
Opening angle 50°

Ultrasonic Sensors  
12x, Opening angle 120°

360°-Camera  
4 Single Cameras  
Opening angle 180°

Driver Camera

Moisture Sensor

Redundant Brake and Steering System



## Example of h (Level 4+/5)

Application for privately-owned vehicles

Testing of Level 4  
automated driving in  
Beijing.







## Examples of current driving functions (Level 4+/5)

### 5th Gen (introduced 2020)

1x 360° Lidar + 4x Lidar

6x Radar

14x Cameras

8x Ultrasonic Sensors

Long Range Camera + Radar

Perimeter Lidar +  
Peripheral Vision System + Radar

Perimeter Lidar +  
Peripheral Vision System

### 6th Gen (introduced 2024)

1x 360° Lidar + 3x Lidar

6x Radar

13x Cameras

8x Ultrasonic Sensors

360 Lidar + 360 Vision System

Perimeter Lidar +  
Peripheral Vision System

Peripheral Vision System  
+ Radar

Source: Waymo

<https://www.youtube.com/watch?v=hWiSQIfHYsY>



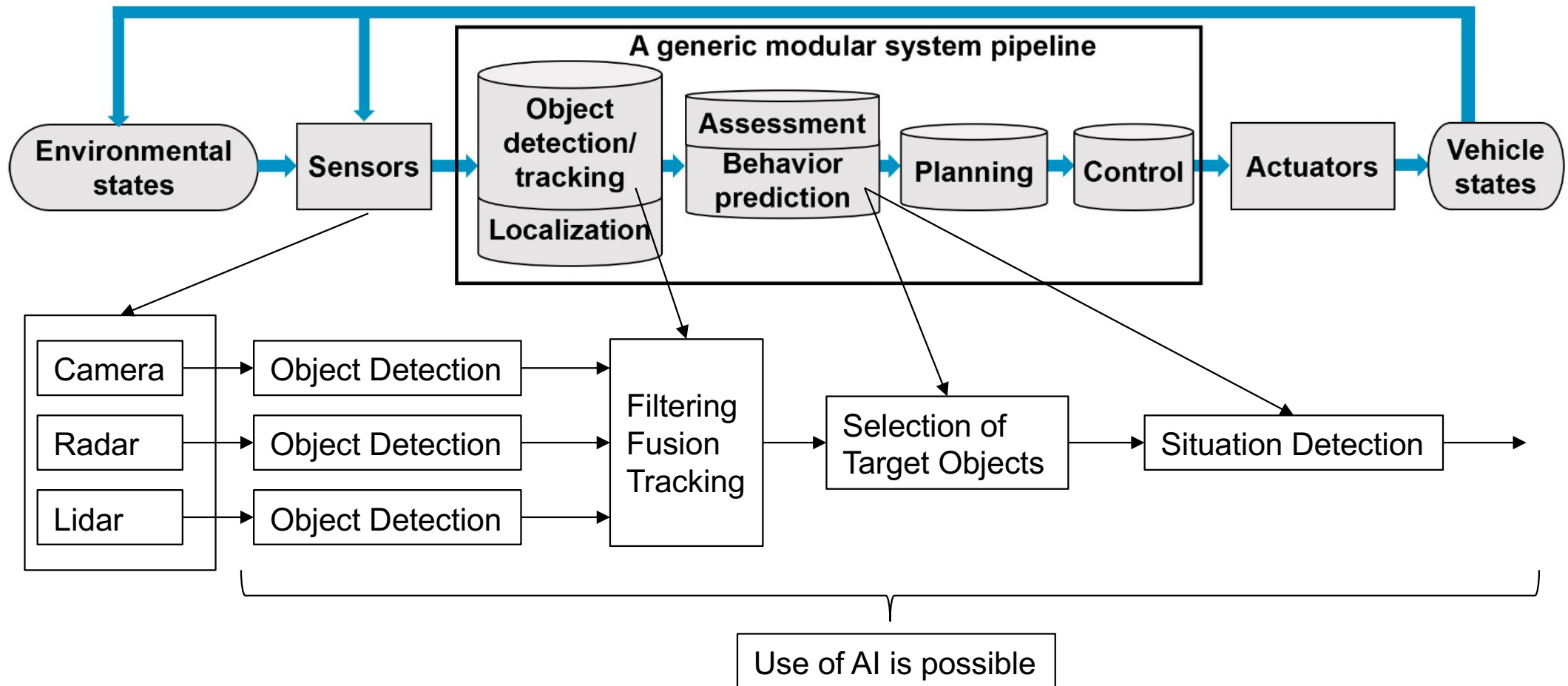


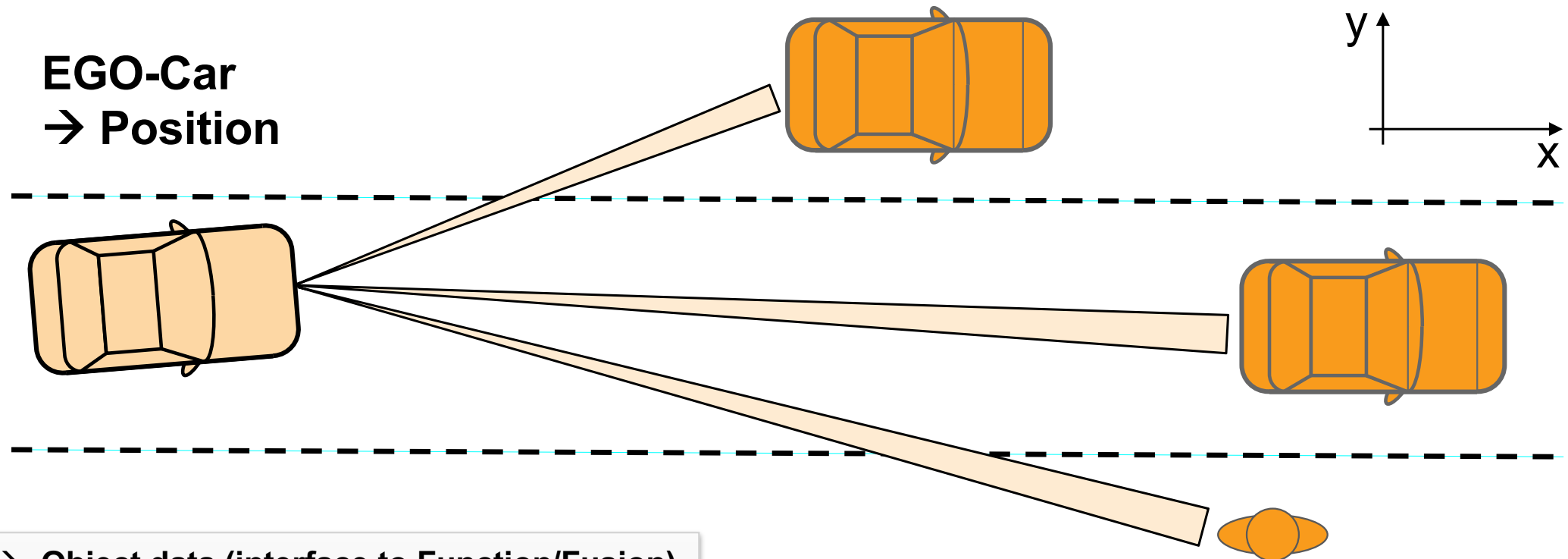
# Surrounding Sensors for Autonomous Driving



# Sensor for autonomous vehicles (1)

Information flow of a generic modular vehicle system:

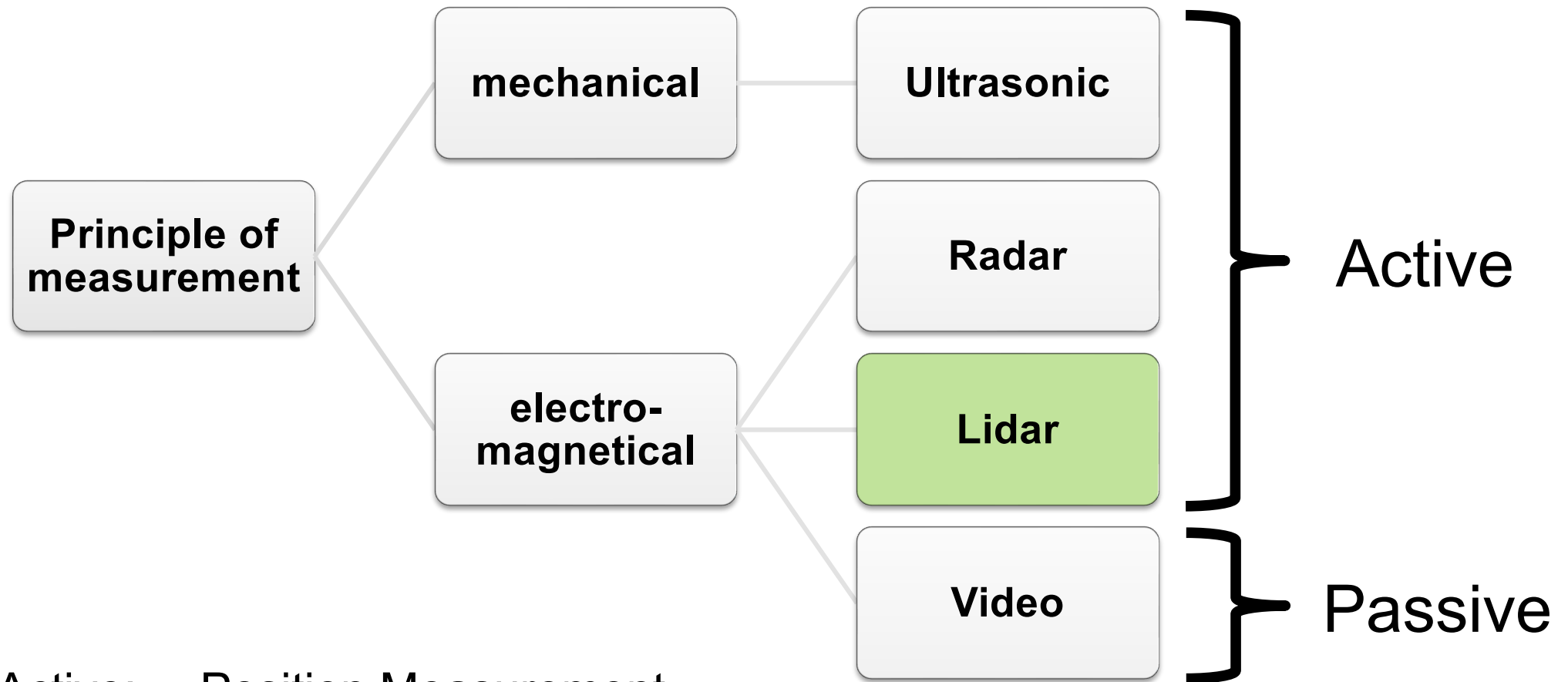




- Object data (interface to Function/Fusion)
  - Different information
  - Quality of measurement
- Situations can be very complex
- Only some situations can be tested !!!

- Surrounding objects**
- Position absolute oder relative to EGO
  - Interpretation depends on situation





Active: Position Measurement

Passive: **Position Estimation!**

## Sensor for autonomous vehicles (4)

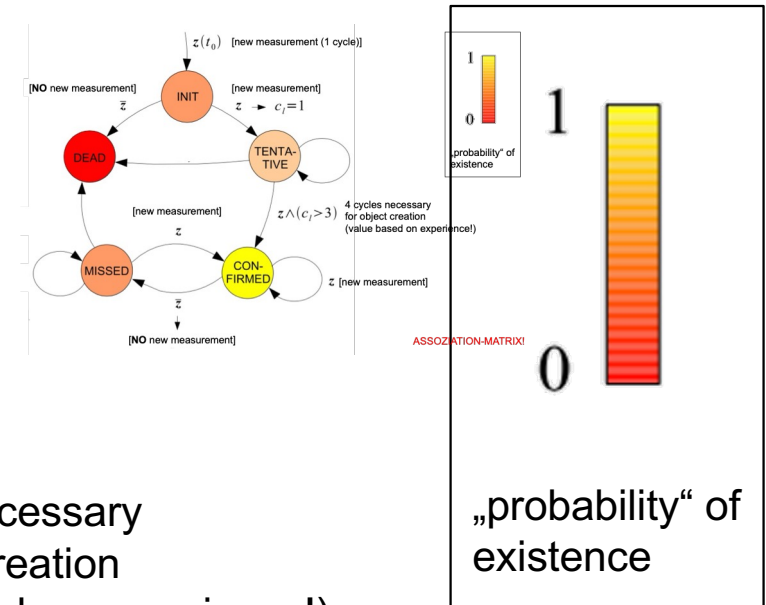
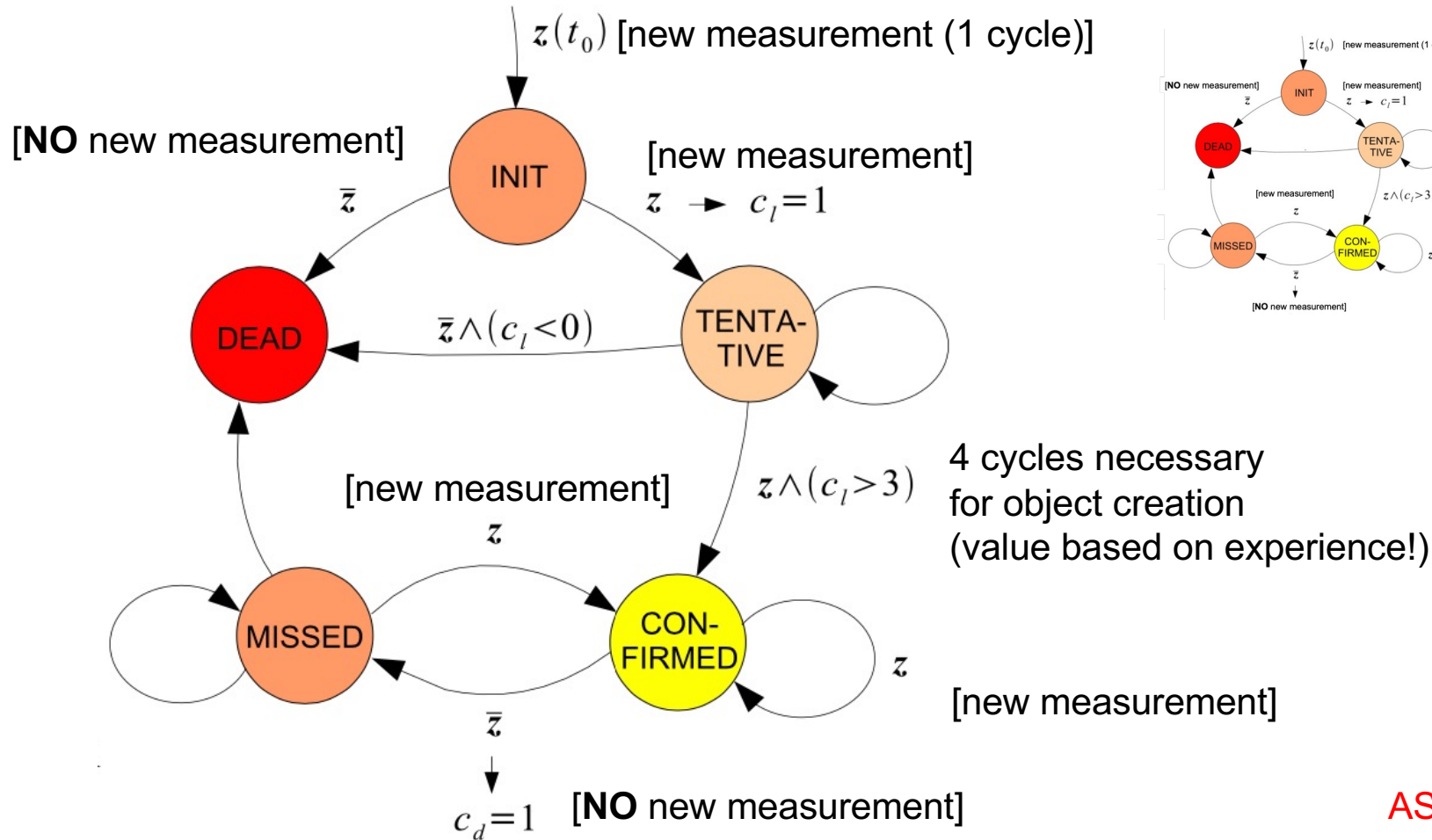
### Comparison:

| Modality                  | Affected by<br>Illumination | Affected by<br>weather | Color | Depth | Range              | Accuracy | Size     | Cost   |
|---------------------------|-----------------------------|------------------------|-------|-------|--------------------|----------|----------|--------|
| Lidar                     | -                           | ✓                      | -     | ✓     | medium<br>(< 200m) | high     | large*   | high*  |
| Radar                     | -                           | -                      | -     | ✓     | high               | medium   | small    | medium |
| Ultrasonic                | -                           | -                      | -     | ✓     | short              | low      | small    | low    |
| Camera                    | ✓                           | ✓                      | ✓     | -     | -                  | -        | smallest | lowest |
| Stereo Camera             | ✓                           | ✓                      | ✓     | ✓     | medium<br>(< 100m) | low      | medium   | low    |
| Flash Camera [77]         | ✓                           | ✓                      | ✓     | ✓     | medium<br>(< 100m) | low      | medium   | low    |
| Event Camera [78]         | limited                     | ✓                      | -     | -     | -                  | -        | smallest | low    |
| Thermal Camera [79], [80] | -                           | ✓                      | -     | -     | -                  | -        | smallest | low    |

\* Cost, size and weight of lidars started to decrease recently [81]

- Use of AI:
- Camera is particularly well suited because it is also used in a variety of other areas,
  - Lidar is simple possible, e.g. by calculating a 2D image (loss of information!),
  - little used for Radar because no clear patterns are recognizable.

Object hypothesis (several measurements/ object fusion):



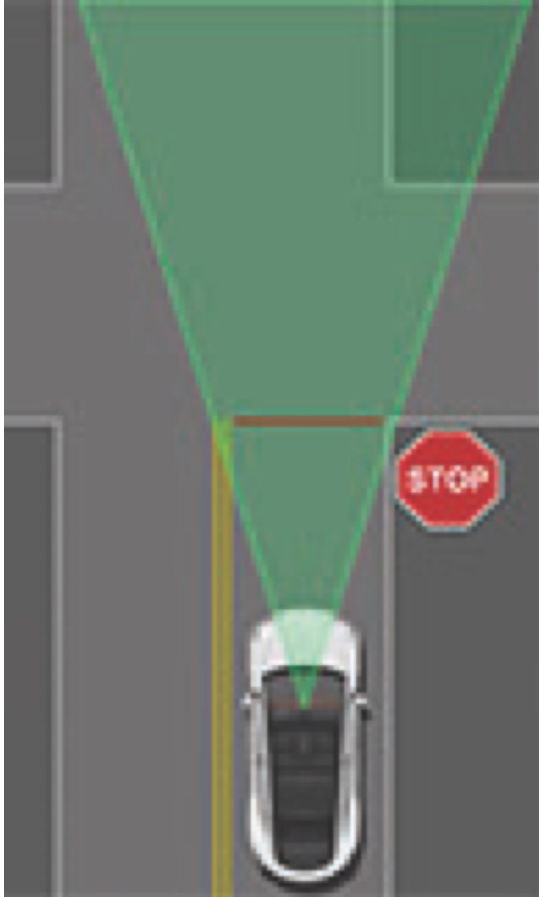
**ASOZIATION-MATRIX!**





# LiDAR in Detail

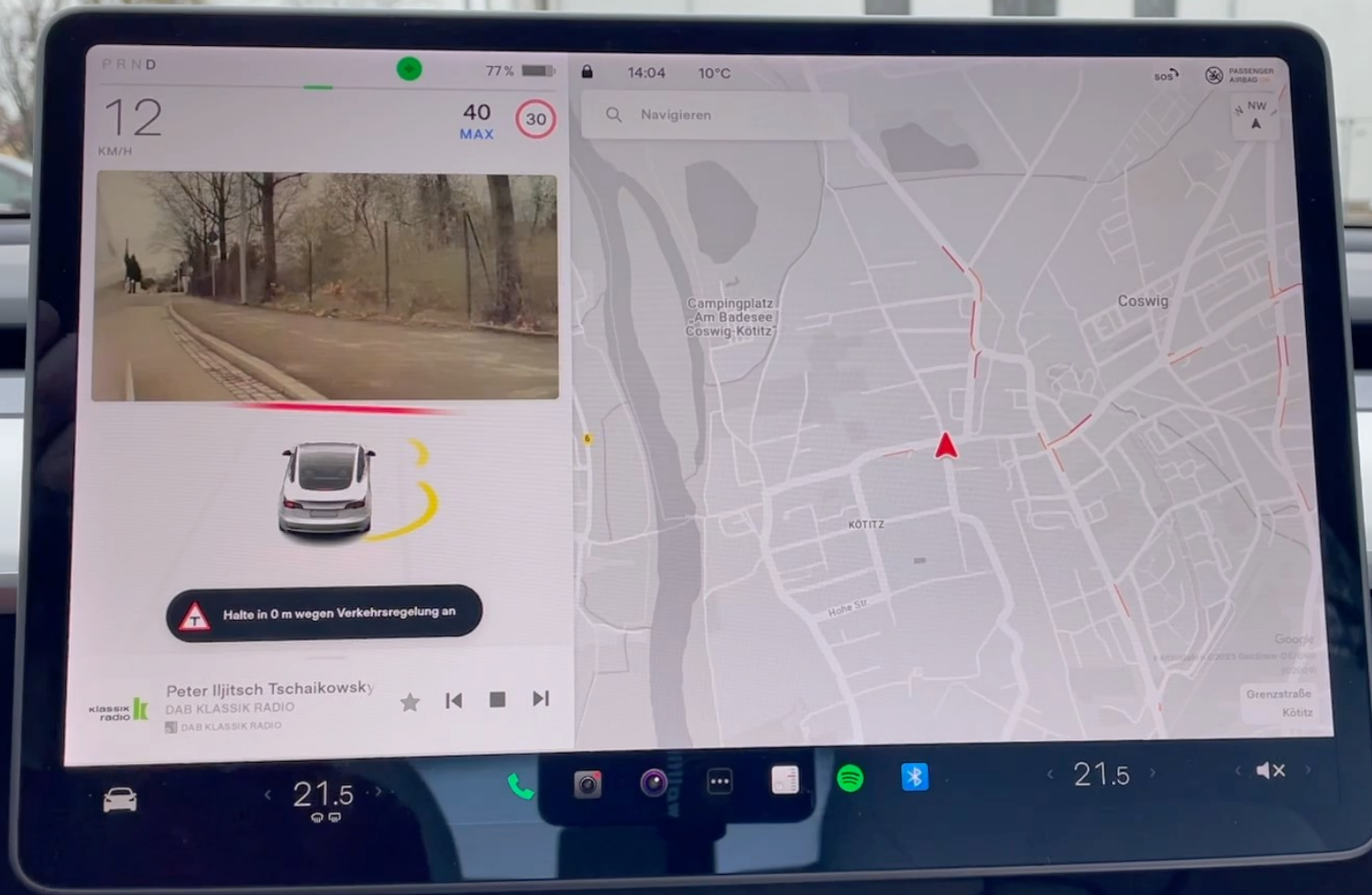




Traffic Light and Stop Sign Assist is a **BETA feature** and works best on roads **frequented by Tesla vehicles**. The traffic light and stop sign assistant tries to stop at all traffic lights and may also stop at green lights.

NOTE: If you confirm that you want to proceed through an intersection with a stop sign by depressing the traction lever or momentarily pressing the accelerator pedal before Model 3 has stopped, your confirmation will be ignored.

Model 3 is not designed to drive past a stop sign without stopping first.



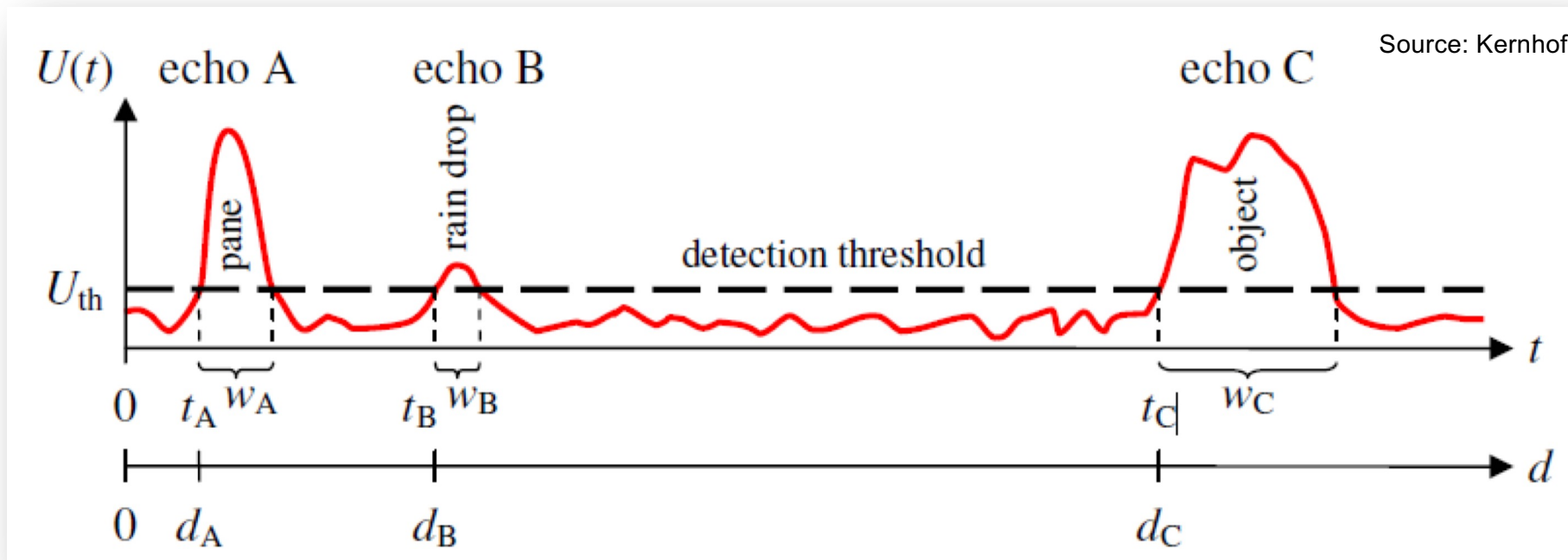


### Different Use Cases

- Detection of road infrastructure
  - Lane Markings
  - Traffic Signs (outline only, no content)
  
- Detection of surrounding objects
  - Vehicles
  - Pedestrians
  
- Localization

## LiDAR (1)

LiDAR, an acronym of **L**ight **D**etection **A**nd **R**anging, is a method for determining ranges by targeting an object or a surface with a laser and measuring the time for the reflected light to return to the receiver (Time-of-Flight – ToF). Lidar may operate in a fixed direction (e.g., vertical) or it may scan directions, in a special combination of 3D scanning and laser scanning. LiDAR sensors for vehicle environment perception use infrared radiation (IR) in the spectral ranges of 850 nm – 940 nm and 1550 nm – 1570 nm (both near-IR ranges) as an excitation source.



## LiDAR (2)

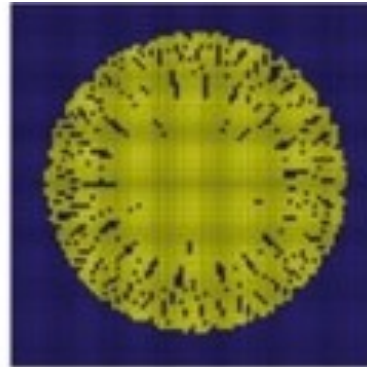


Livox Horizon  
81.7° FOV

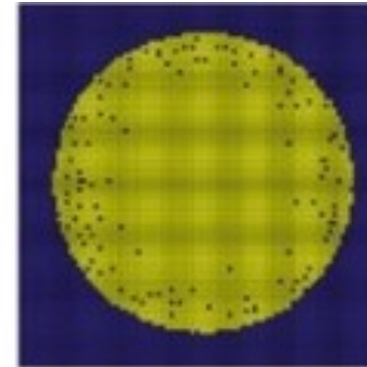
1. Cycle



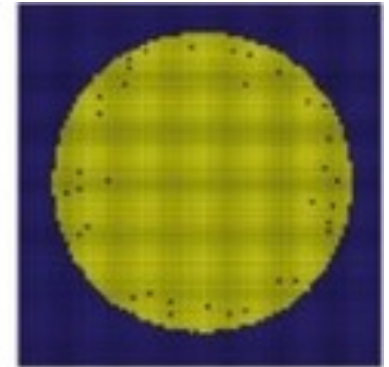
2. Cycle



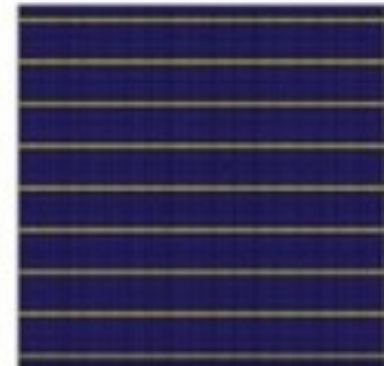
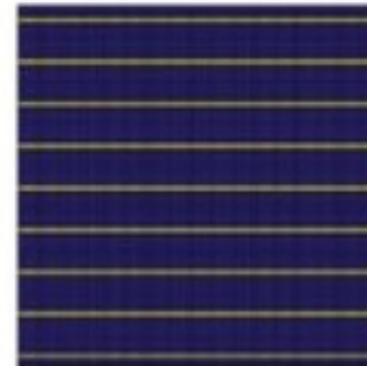
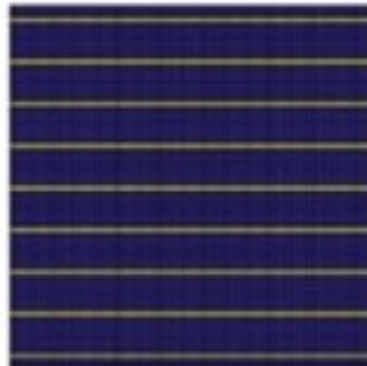
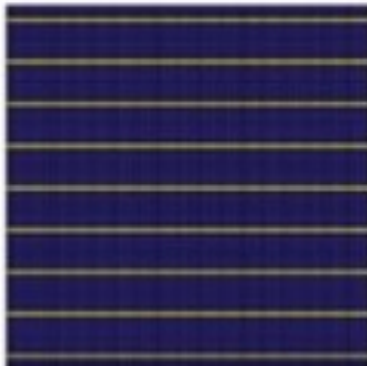
3. Cycle



4. Cycle



Ouster OS1-64  
360°, 64 Layers





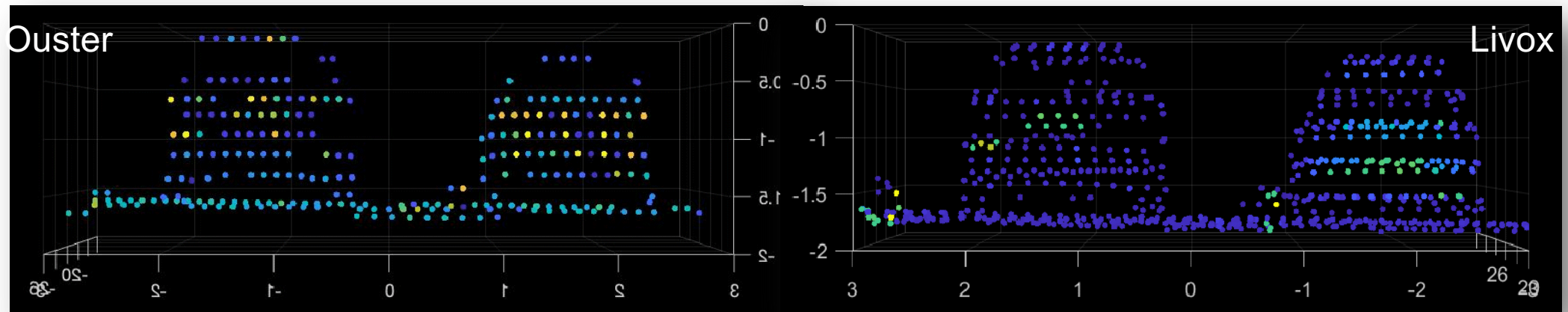
## LiDAR (3)



Ouster OS1-64  
360°, 64 Layers



Livox Horizon  
81.7° FOV



Range is highly dependent on the reflectivity of the objects.

Example Livox: **90 m @ 10% reflectivity**  
**130 m @ 20% reflectivity**  
**260 m @ 80% reflectivity**

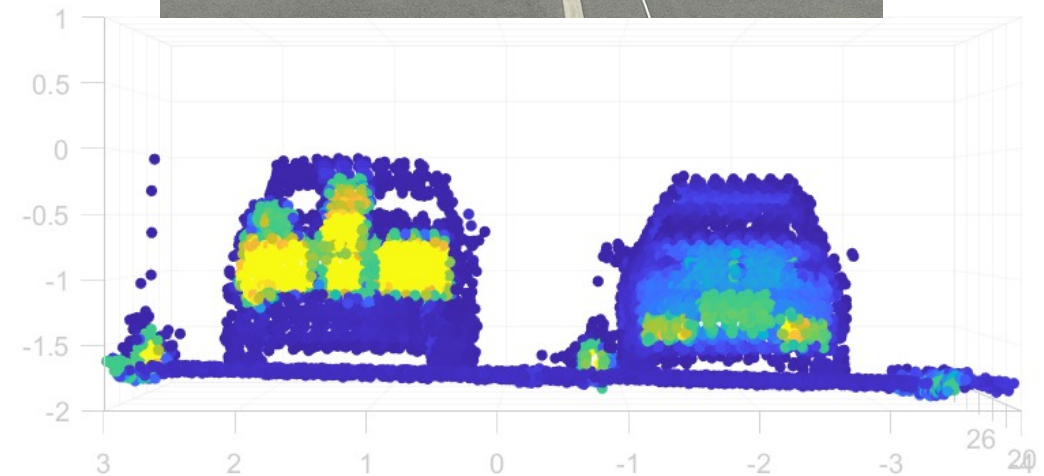
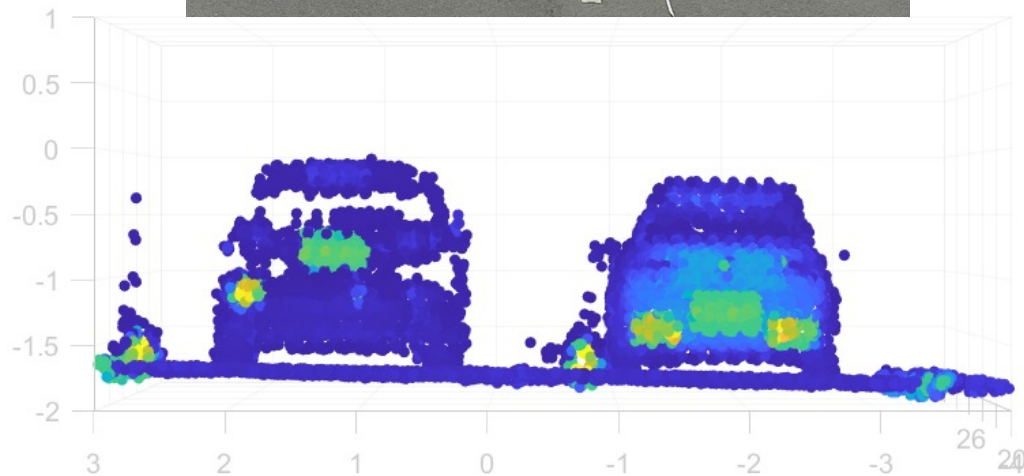
*„The LiDAR sensor may fail to recognize obstacles and thus Autonomous Emergency Brake (AEB) may fail to operate as intended in some situations, including but not limited to:*

- In rain, snow, fog, haze, and other bad weather.*
- The sensing performance is poor due to exhaust gas, splashes, snow, or dust kicked up by vehicles in front.*
- The vehicle is driving on wet roads or roads with water.*
- False warnings may be generated due to traffic signs and anti-collision buckets on expressways and elevated roads.“*

Source: NIO ET7 User Manual

## LiDAR (5)

2 Cars with different colours @ 20m → 1.0 second measurement





### WHY ARE ROAD MARKINGS IMPORTANT?

Keeping road markings in visible and understandable condition is vital for road safety.

They help road users (**and systems !**) to:

- navigate the road systems,
- making us aware of upcoming hazards,
- help us drive appropriately for the situation.

Without their clarity, it can cause confusion on the roads.

Road markings are especially important at nighttime when, in the dark conditions, we rely on their presence even more so.

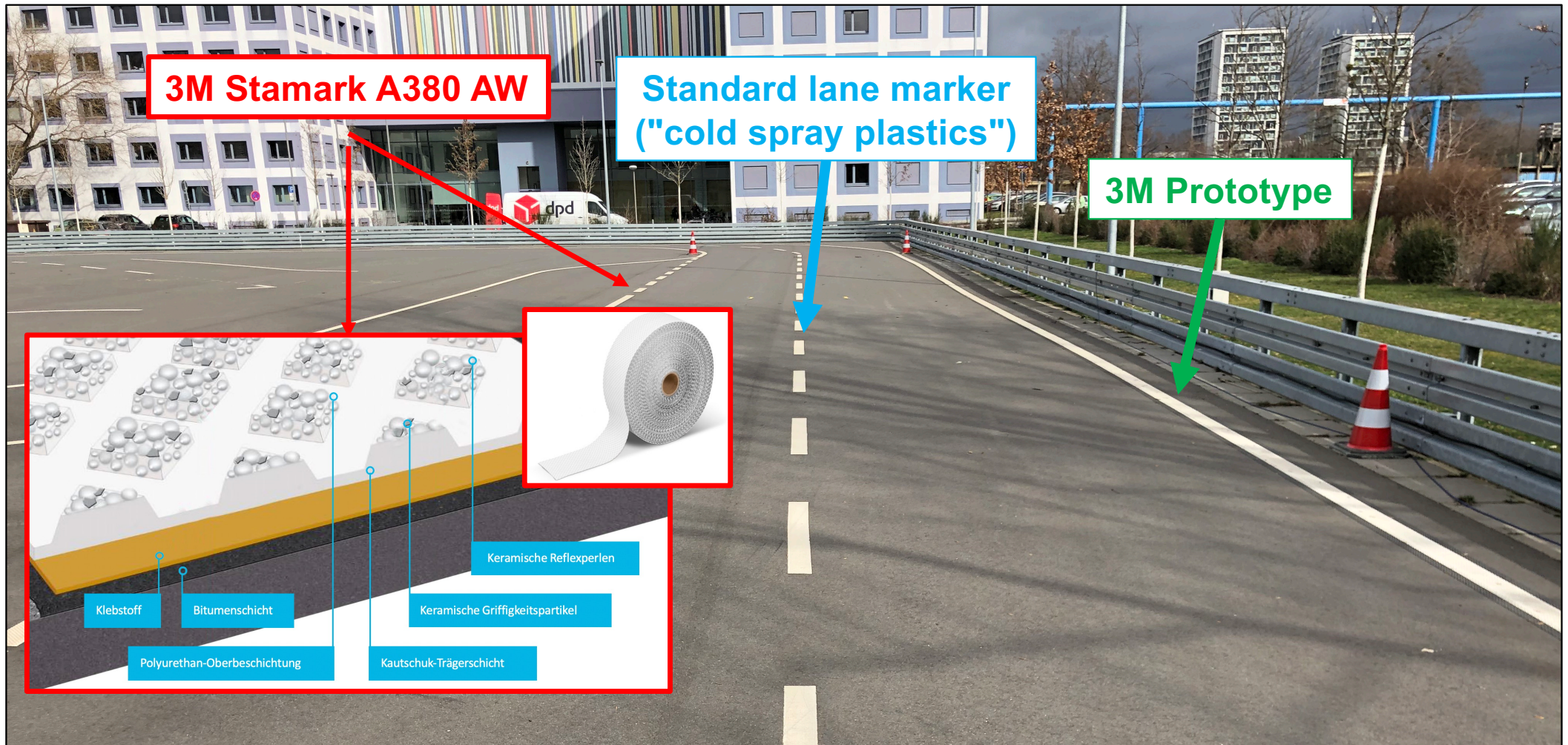
Source: [www.threecountieslining.co.uk](http://www.threecountieslining.co.uk) (processed)

→ Retroreflexion durch neue Fahrspurmarkierungen (z.B. Fa. 3M)

Simple Threshold Detection: **Intensity**<sub>LID</sub> >  $\lambda_{UG}$

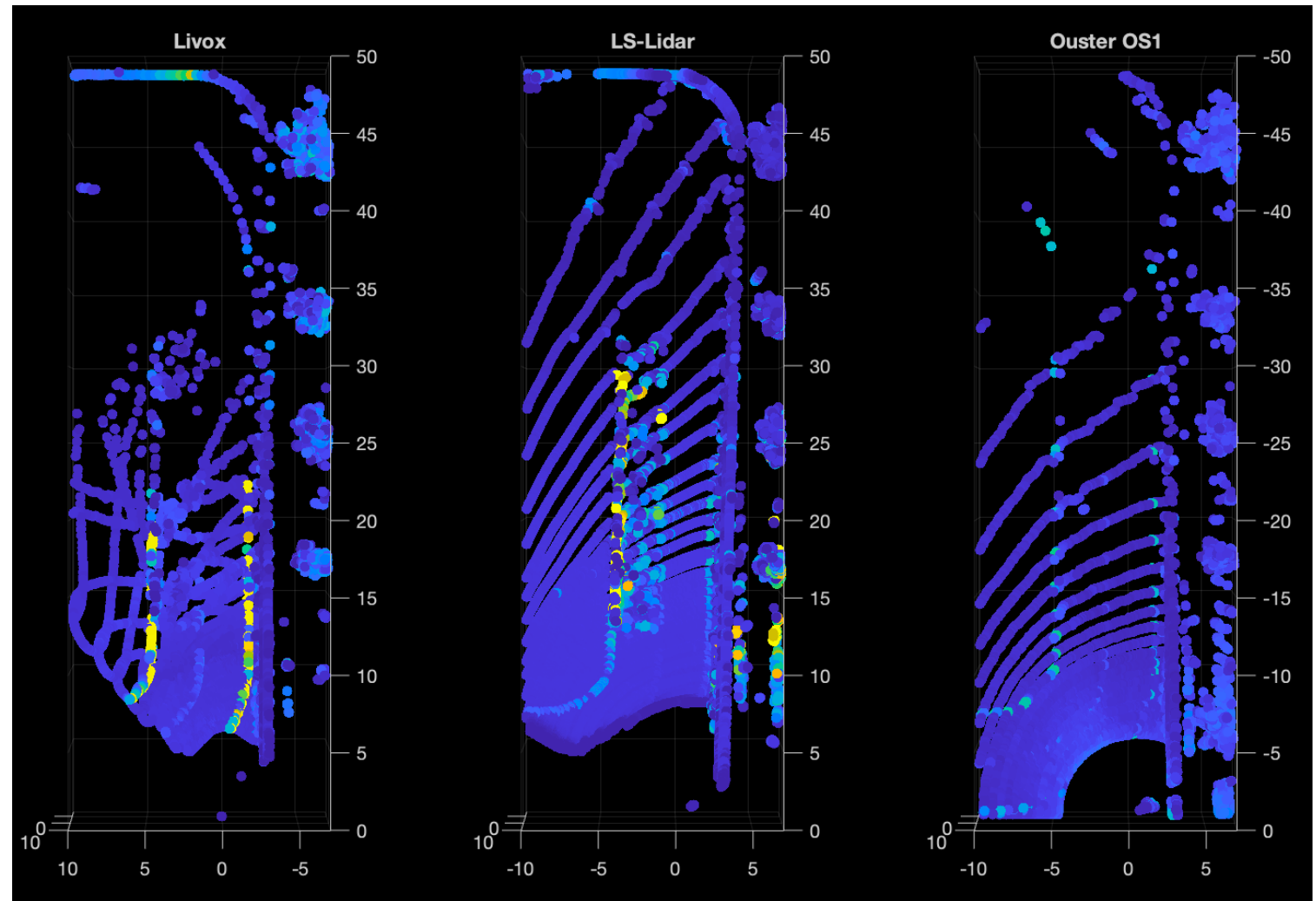
Measurement → Threshold → Curvefitting

## Lane Markings Detection (2)



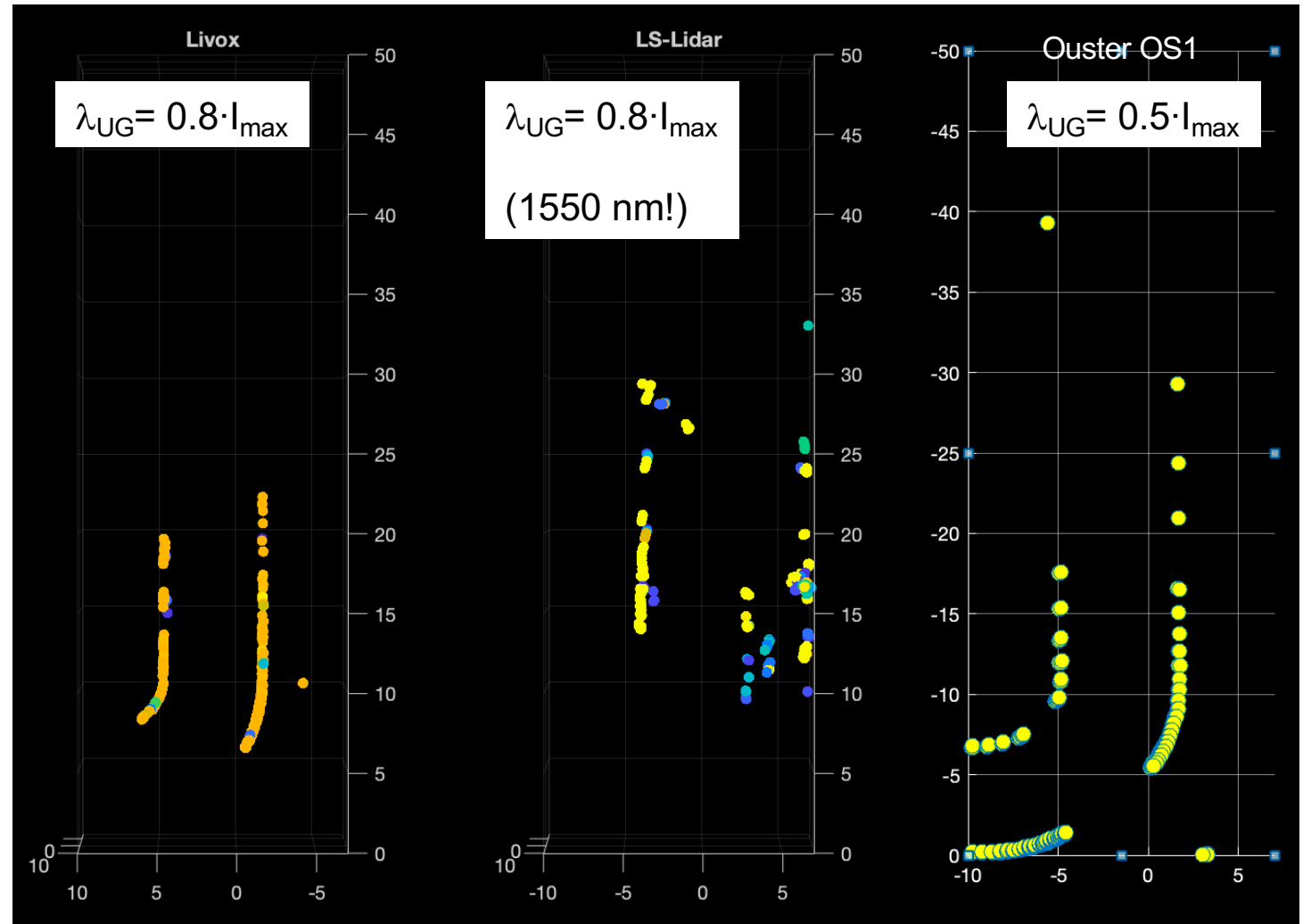
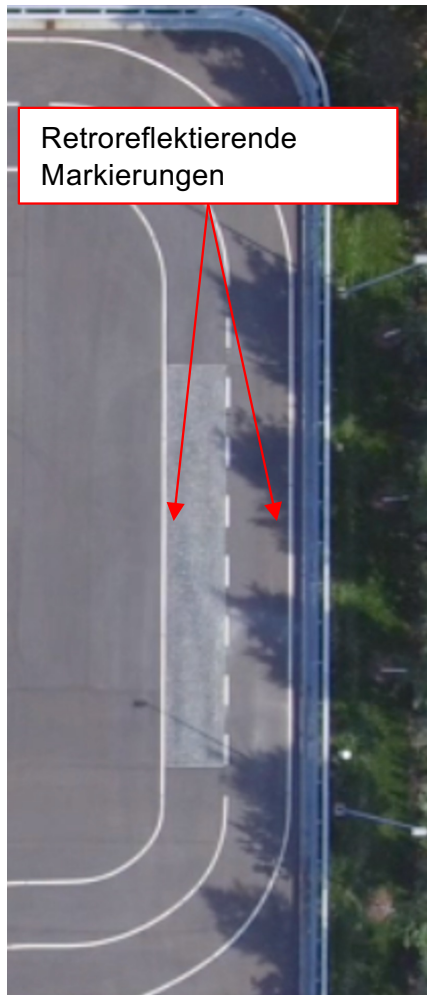


## Lane Markings Detection (3)





## Lane Markings Detection (4)



## Bounding Box Based Object Detection (1)

Fits a plane to a point cloud that has a maximum allowable distance from an inlier point to the plane. The function returns a geometrical model that describes the plane. This function uses the M-estimator Sample Consensus (MSAC) algorithm to find the plane (variant of the RANDOM Sample Consensus (RANSAC) algorithm).

↓

```
[model,inliers,outliers] = pcfitplane(ptCloudIn, maxDistance);
```



Parameter of Distance

```
ptCloudWithoutGround = select(ptCloudIn, outliers);
```

```
[labels,numClusters] = pcsegdist(ptCloudWithoutGround, distThreshold);
```



Parameter of Distance

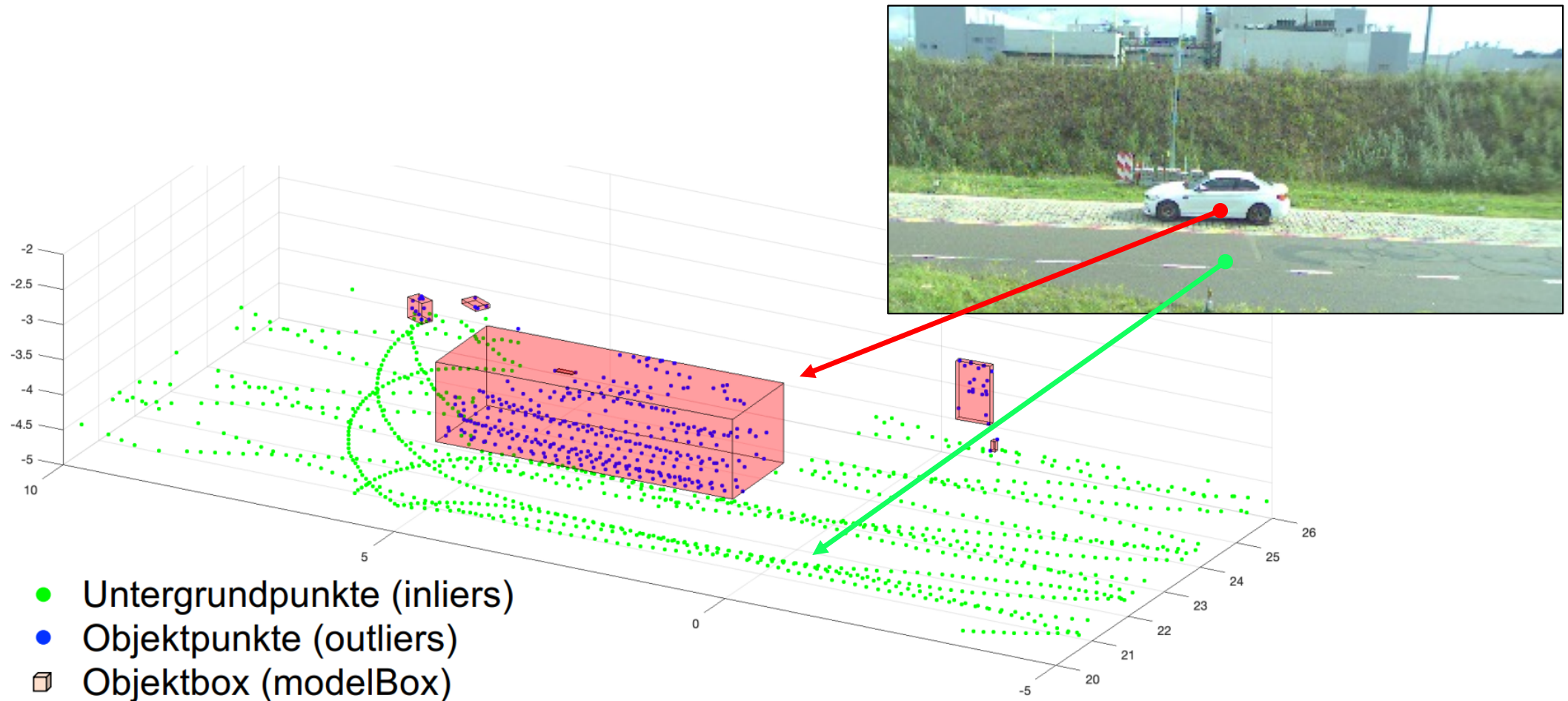
```
for i = 1:numClusters
```

```
    idx = find(labels == i);
```

```
    modelBox = pcfitcuboid(ptCloudWithoutGround, idx);
```

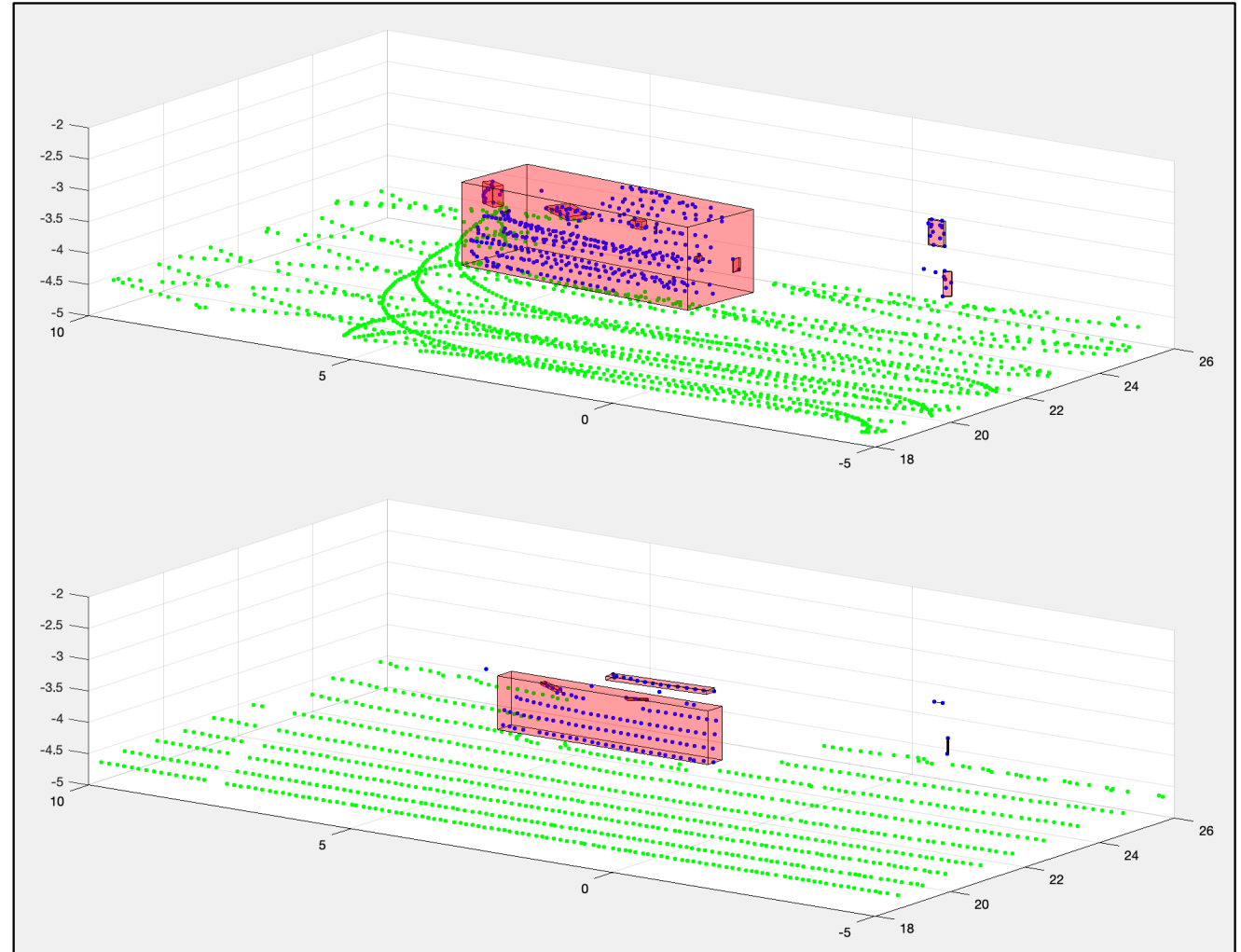
```
end
```

## Bounding Box Based Object Detection (2)

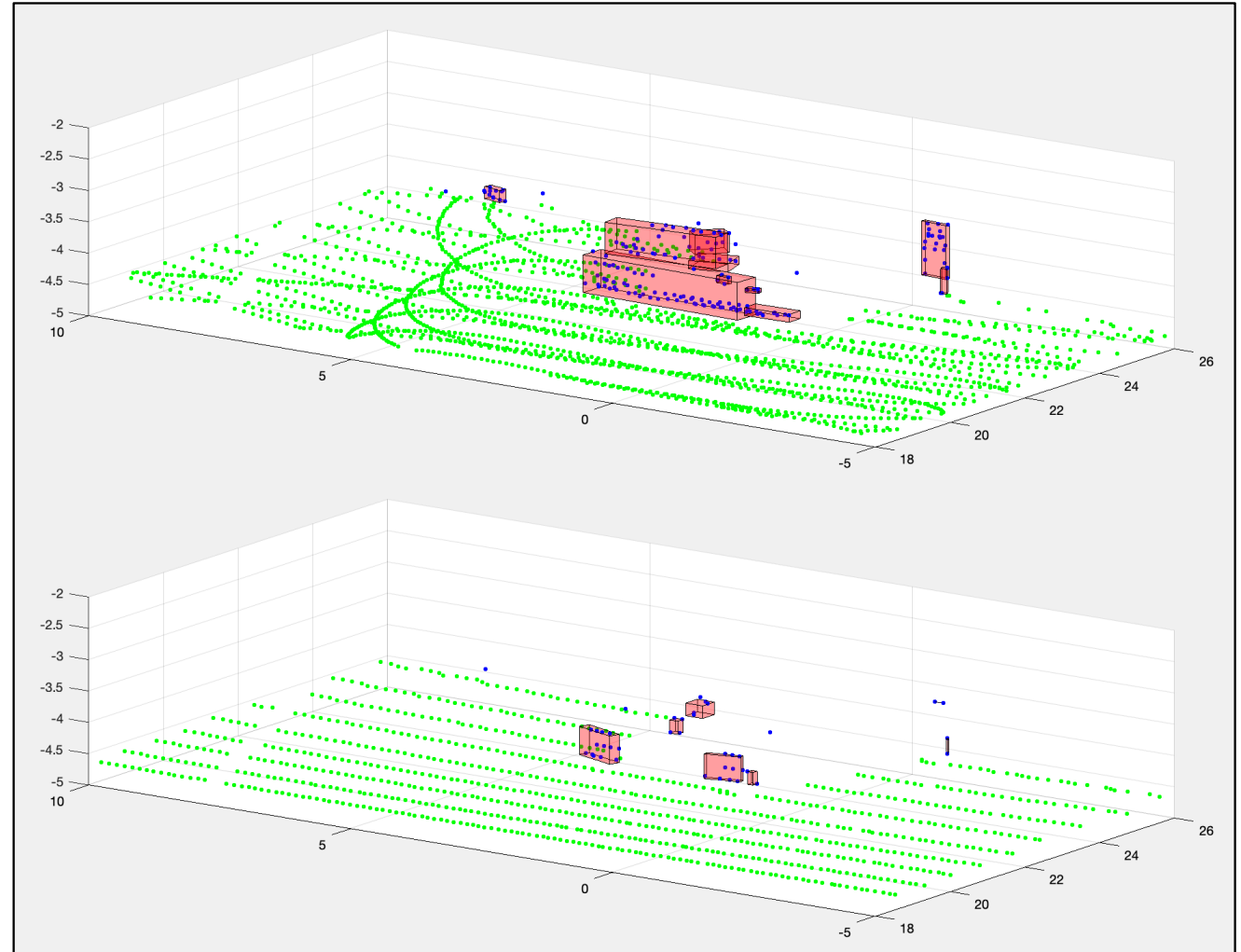




## Bounding Box Based Object Detection (3)

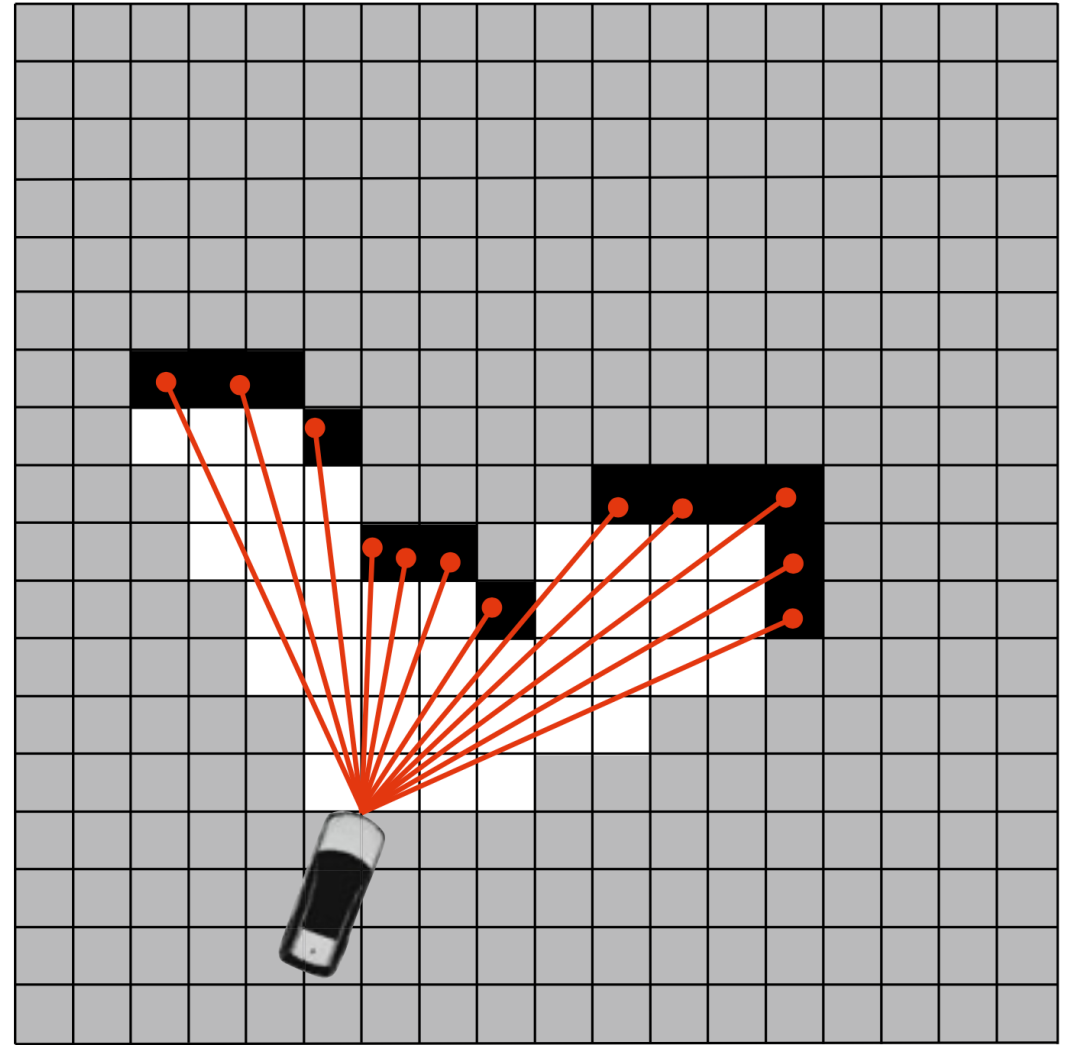


## Bounding Box Based Object Detection (4)



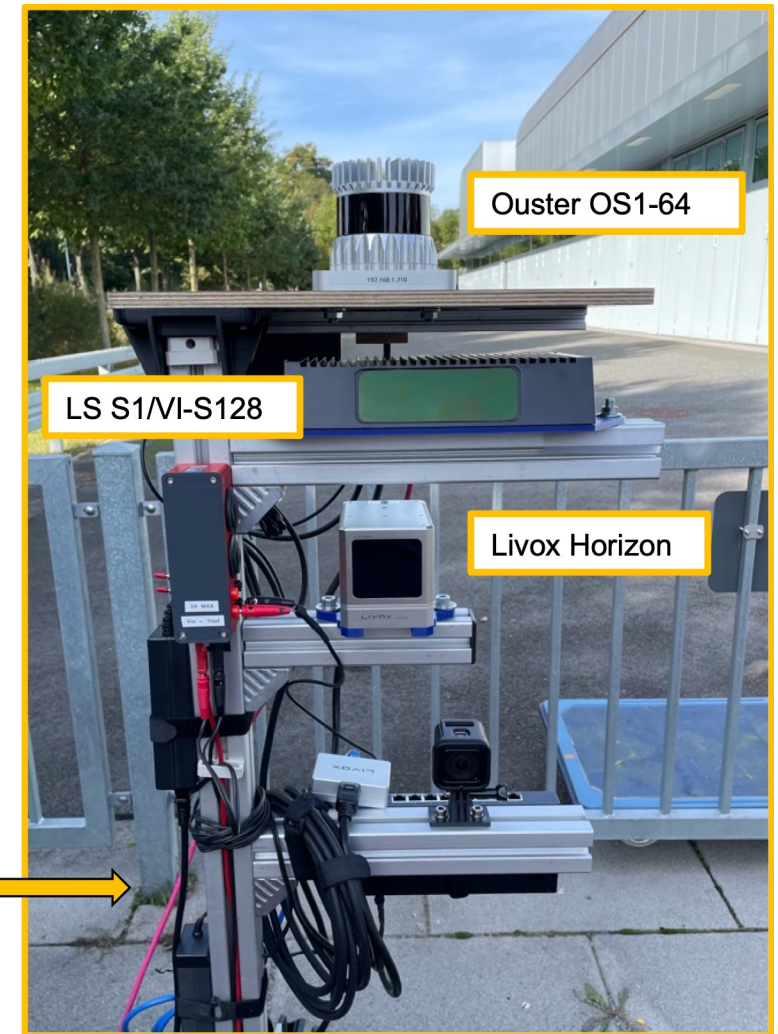
## Grid-based Object Detection (1)

Grid maps divide the vehicle's environment into cells. Each of these cells represents a location about which it contains information. When sensor data from various sensors allow inferences to be drawn regarding the state of the cells, the mapping process in grid maps also represents an indirect form of sensor data fusion. The basic concept can be illustrated by the example shown alongside. LiDAR beams are reflected by obstacles, allowing occupancy in the corresponding cells to be inferred (shown in black). Cells located between the sensor and the reflection points are assumed to be unoccupied (shown in white). This measurement provides no information regarding the occupancy status of other cells (shown in grey).

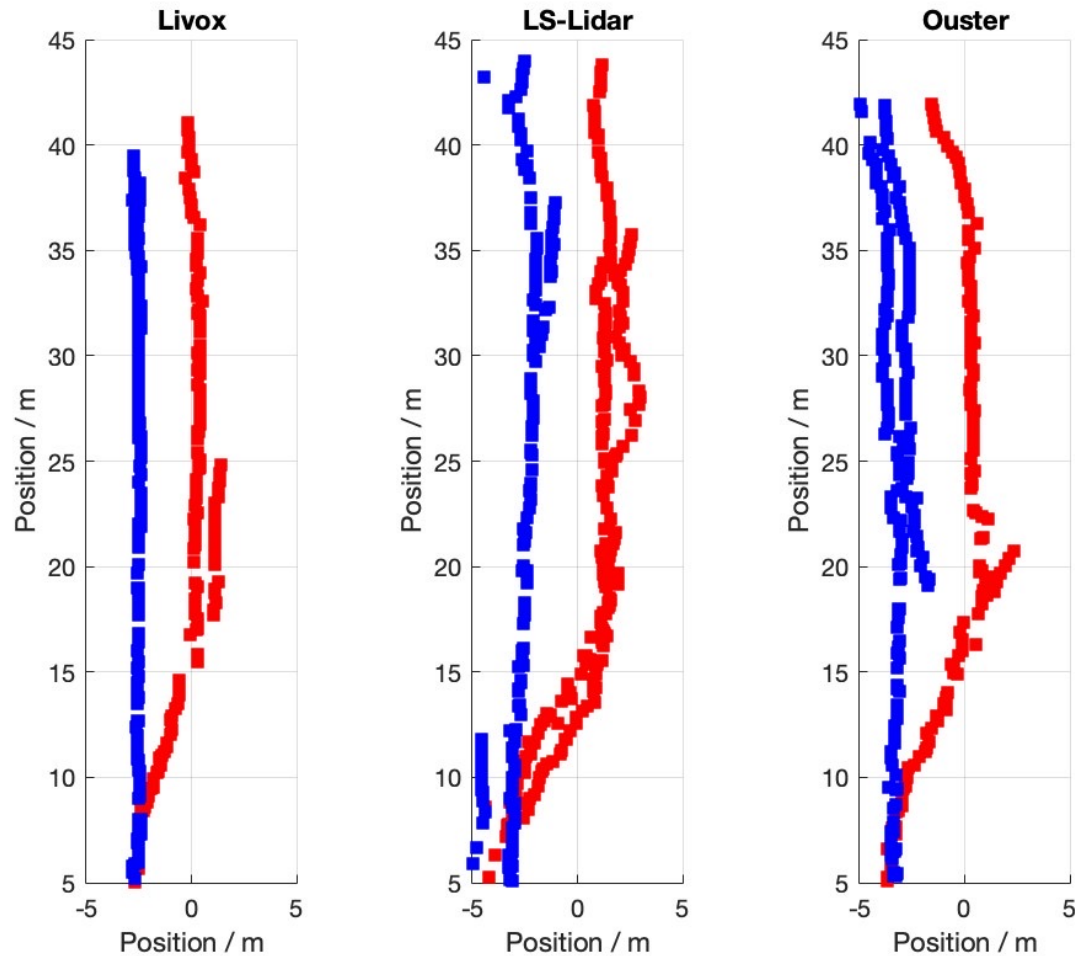




## Grid-based Object Detection (2)

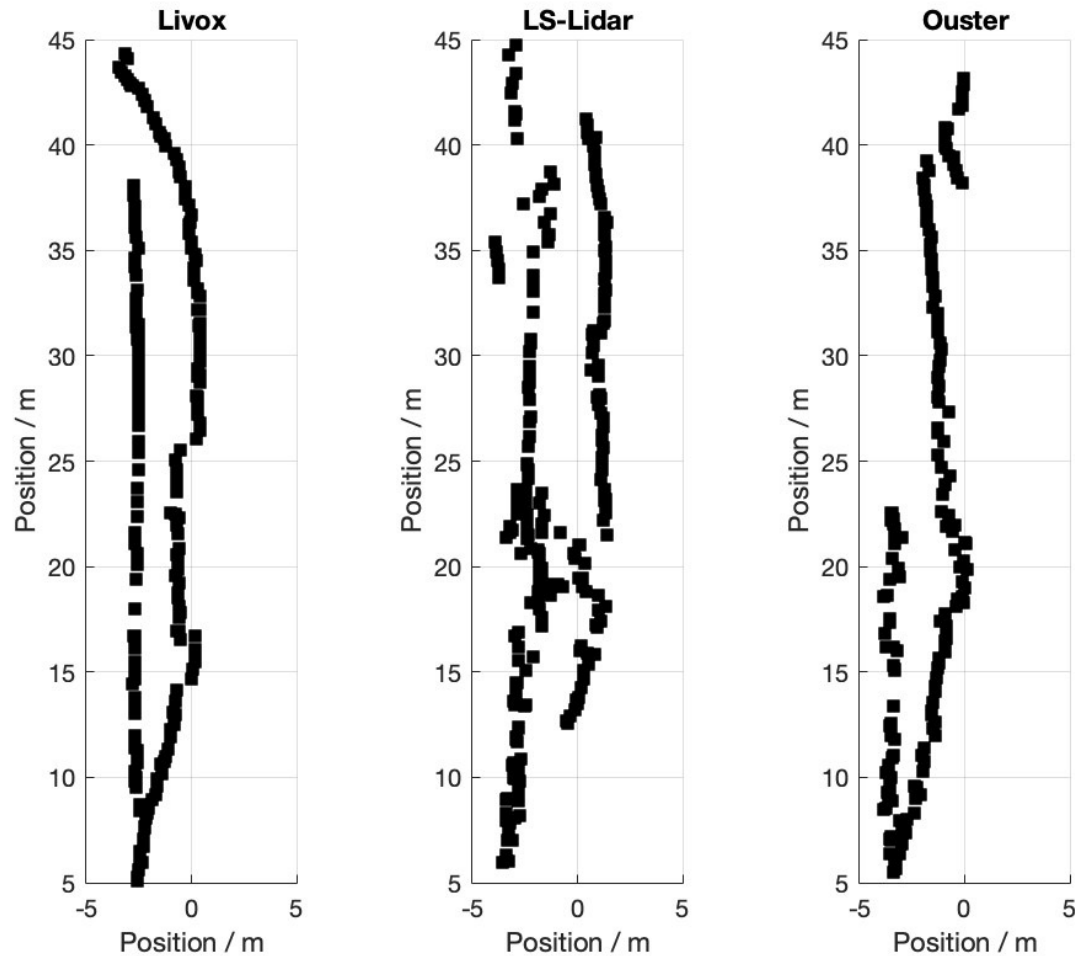


## Grid-based Object Detection (3)



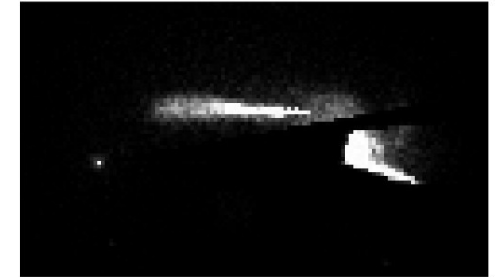
- A (Porsche)
- B (BMW i3)

## Grid-based Object Detection (4)



C  
Porsche +  
BMW i3

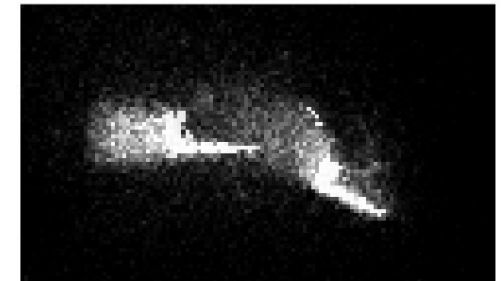
Livox



LS-Lidar



Ouster





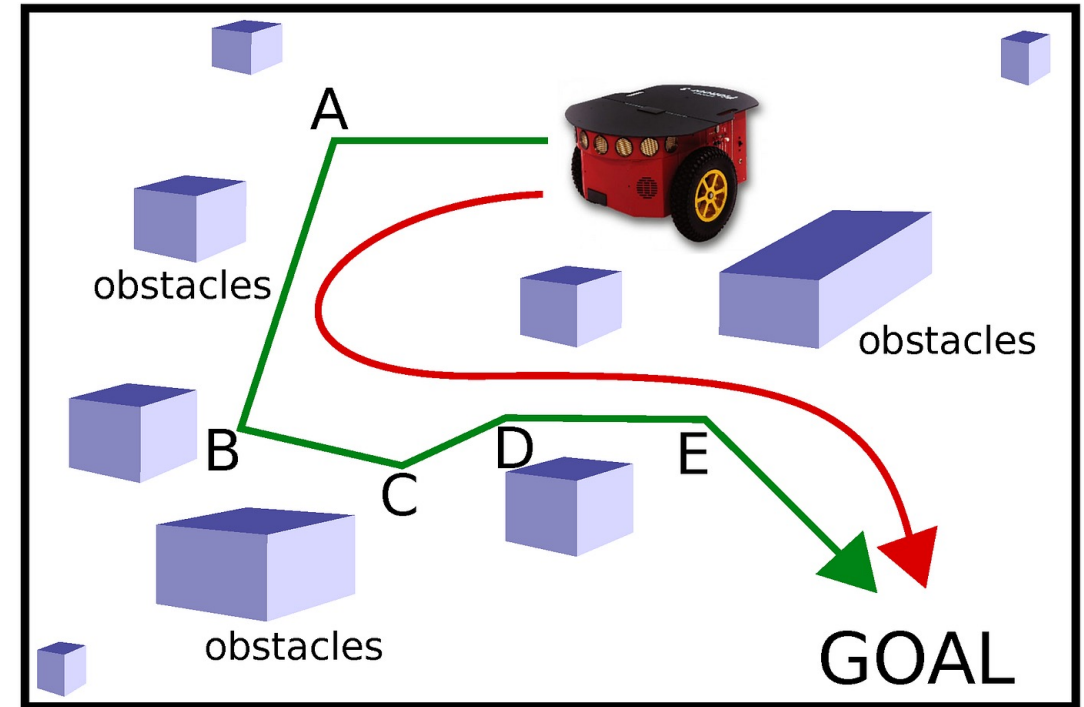
# Localization (1)

A robot (and an autonomous car) only takes information about its surroundings from its sensors, and it needs to use that information to successfully complete the task it is given. To do that, it needs to understand two basic things:

- **Where** it is at the moment, and what is its **orientation** (*Localization*)
- What and where are the **objects around it**, and how do they affect its actions (*Mapping*)

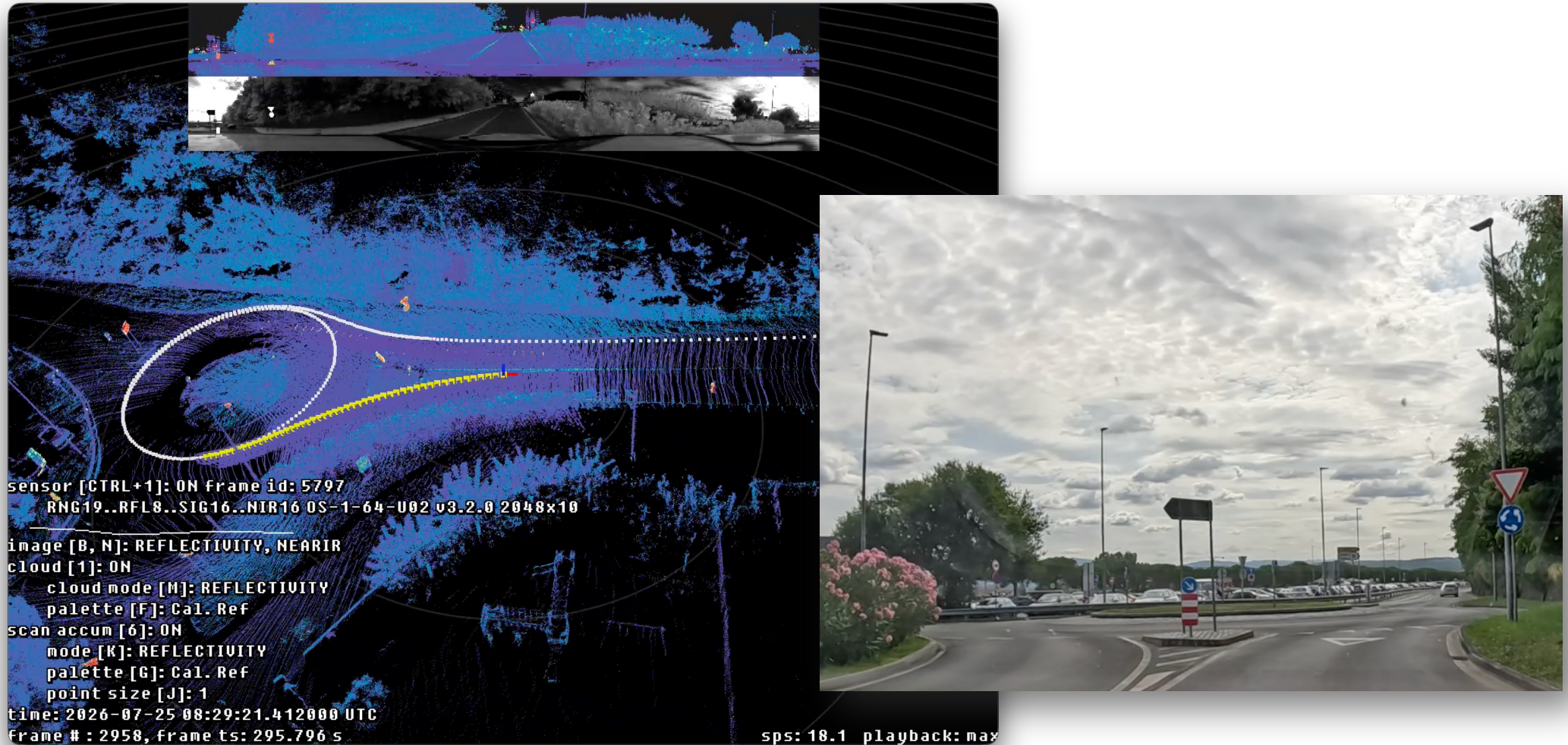
The desired solution to the above problem is SLAM: an algorithmic approach to enabling autonomous robots to both create a map of the surrounding area, using sensors, and use that map to navigate the area at the same time.

SLAM essentially implements the same process that humans do, using the robot's sensors and processor. This is a “chicken-or-egg” problem: to determine its location it needs a map, and to determine the map we need the object locations.



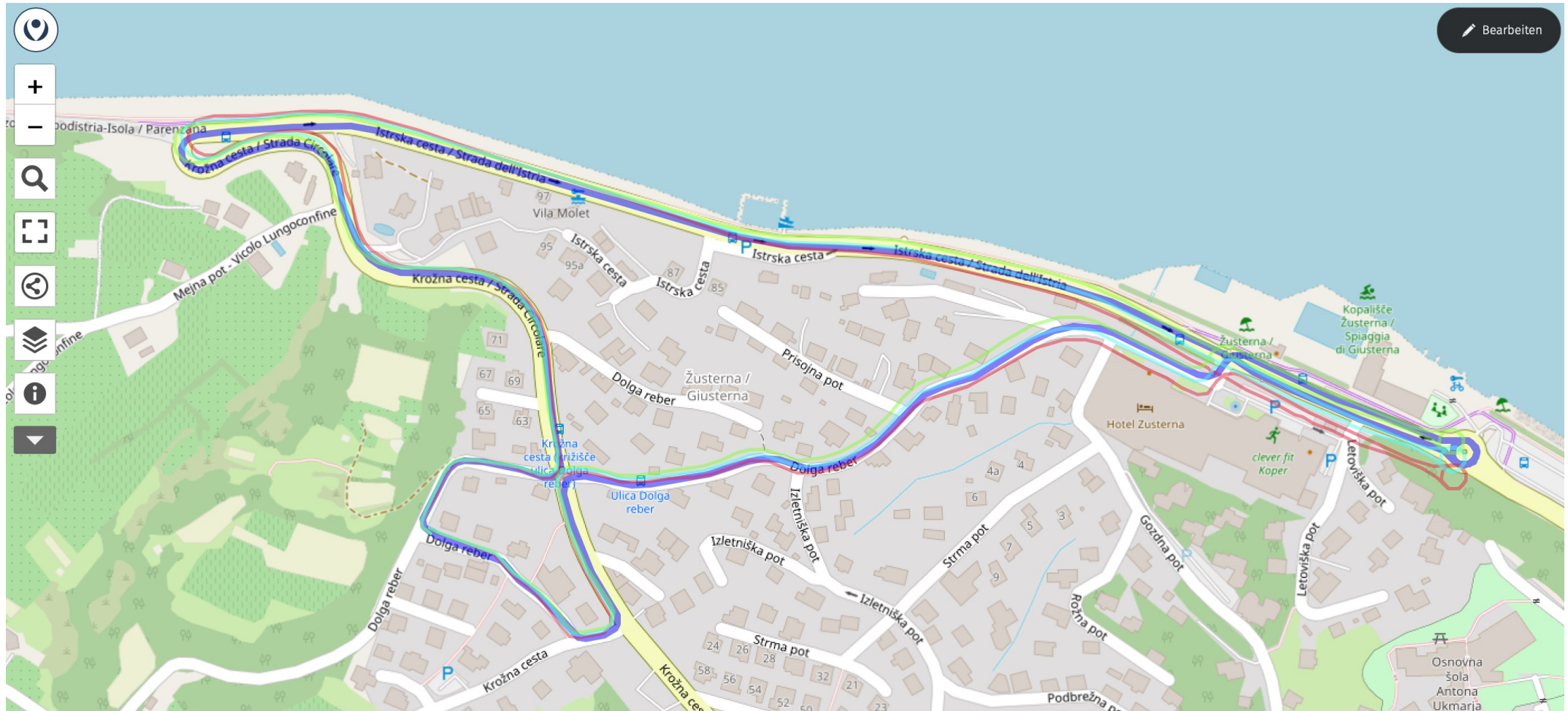
<https://medium.com/@pliam1105/using-slam-to-autonomously-navigate-in-unknown-environments-01952667b34a>

## Localization (2)





## Localization (3)



[https://umap.openstreetmap.de/de/map/testmessung-koper-ouster-slam\\_148093#17/45.546019/13.705369](https://umap.openstreetmap.de/de/map/testmessung-koper-ouster-slam_148093#17/45.546019/13.705369)





# AI for LiDAR



Using AI for these sensors is not as straightforward as with image processing. There are many more parameters that influence the result. Even small changes in position can have a significant impact on the outcome.

However, the basic structure of image processing can be used with lidar. For this, the 3D image must be converted into a 2D image based on the distance. The classification for the learning process must then be applied to this type of image; existing models cannot be used. Since different lidar systems generate different patterns for the same object, the results are difficult to transfer. The effort is therefore currently too high, and no standard exists.

The LiDAR from Ouster Inc. (USA) is particularly well suited for this purpose, as it directly generates various 2D images:

- Reflectivity
- Near IR (Ambient)
- Range
- Signal

→ [Show LiveData | Measurement](#) (auch auf vorherige Segmentierung und fehlende Schätzung Fahrbahnebene eingehen)

**Reflectivity:** This is a **calibrated** 8-bit value that represents the reflectivity of the surface or object detected by the sensor. It is scaled based on the measured range and the sensor's sensitivity at that range, providing a consistent indication of the target's reflectivity across different lighting conditions and distances. Reflectivity is useful for material classification and is independent of the distance to the object or ambient light levels.

**Near-IR (Near Infrared):** This is a 16-bit value representing the strength of ambient sunlight (natural environmental illumination) detected at each point, expressed as the quantity of photons collected that were not produced by the sensor's laser pulse. Near-IR data varies with sunlight levels and is not available at night or indoors. It provides information about the ambient light conditions in the scene.

**Range:** This is a 19-bit (or 15-bit in low data rate mode) value representing the distance to the detected object, measured in millimetres. The range is discretized to the nearest millimetre, with a maximum detectable range of up to 524 meters (for 19-bit mode). A range value of 0 indicates no detection or an out-of-range measurement.

**Signal:** This is a 16-bit value representing the raw number of photons (signal intensity) detected from the laser pulse returned to the sensor. The signal value varies with range—objects farther away return less light, resulting in a lower signal. It is a direct measure of the strength of the returned laser pulse.



**Reflectivity**



**Near-IR**



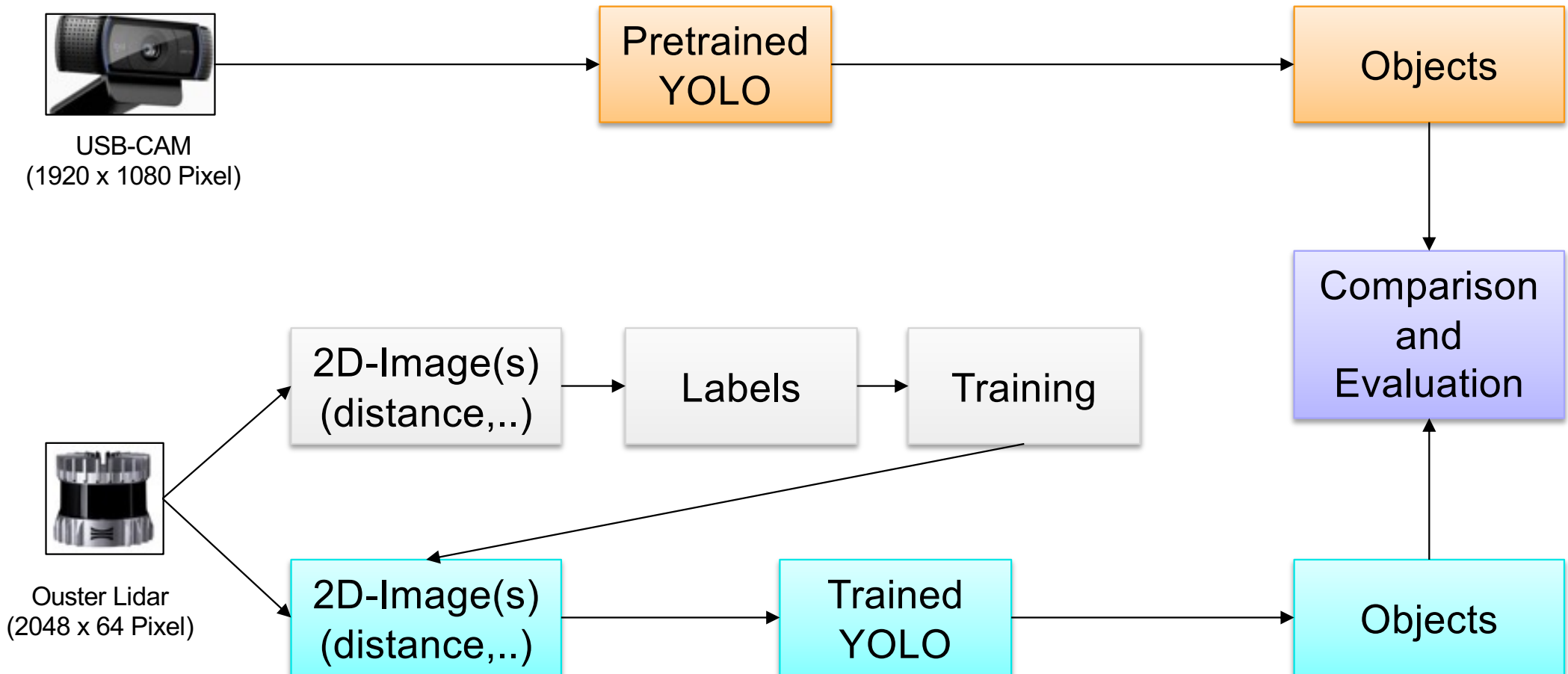
**Range**



**Signal**



Overall process:



- **Purpose:** Evaluate the performance of an AI-based image detection model.
- **Definition:** A confusion matrix is a table that summarizes the performance of a classification model by comparing actual vs. predicted labels.
- **Key Terms:**
  - **True Positive (TP):** Correctly predicted positive class.
  - **True Negative (TN):** Correctly predicted negative class.
  - **False Positive (FP):** Incorrectly predicted positive class.
  - **False Negative (FN):** Incorrectly predicted negative class.

Collect Predictions: Run your AI model on a test dataset and collect the predicted labels.

Gather Ground Truth: Ensure you have the true labels for the test dataset.

Build the Matrix: Populate the confusion matrix by comparing predictions to ground truth:

Rows: Actual classes.

Columns: Predicted classes.

Visualize: Use libraries like `sklearn.metrics.confusion_matrix` (Python) or Matplotlib/Seaborn for visualization.



- **Metrics to Calculate:**
  - **Accuracy:**  $(TP + TN) / \text{Total}$
  - **Precision:**  $TP / (TP + FP)$
  - **Recall (Sensitivity):**  $TP / (TP + FN)$
  - **F1-Score:** Harmonic mean of precision and recall.
- **Error Analysis:**
  - Identify misclassified classes (high FP/FN).
  - Improve model training or data labelling for problematic classes.
- **Example:** Use heatmaps to visualize confusion matrices for multi-class problems.

→ [Show Python](#)

**Thank you for your attention and  
have fun working together.**

Prof. Dr. rer. nat.  
**Toralf Trautmann**  
Faculty Mechanical Engineering

T +49 351 462 2383  
trautmann@htw-dresden.de

