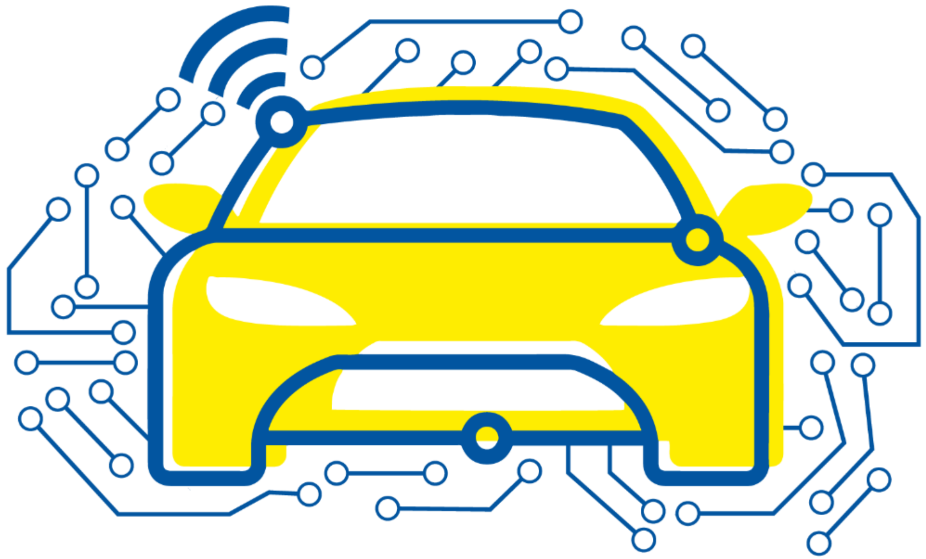
Automated and Connected Driving Challenges

Section 2 – Sensor Data Processing

Object Detection

Training



Bastian Lampe

Institute for Automotive Engineering

Object Detection - Training Pipeline

Data Acquisition

- Large **ground truth data** have to be **available**

- **Using existing reference data**
  - Reference data is from another domain (sensor setup, geolocation) → *domain shift*
  - Reference model is only trained on reference data → no info on *generalization capabilities*
  - Lack of full, variable annotated public datasets

Object Detection - Training Pipeline

Data Acquisition

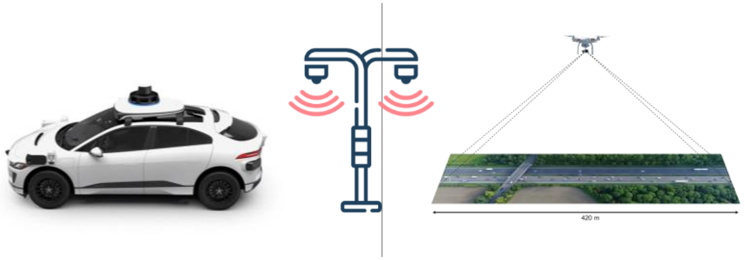
- Large **ground truth data** have to be **available**

- **Using existing reference data**
  - Reference data is from another domain (sensor setup, geolocation) → *domain shift*
  - Reference model is only trained on reference data → no info on *generalization capabilities*
  - Lack of full, variable annotated public datasets

- **Creating own datasets**
  - High **manual effort** for creating own datasets

  - **Simulation** data
  - **Real-world** data: measurement vehicle, infrastructure sensors, drones

  - Labeling approaches
    - Manual labeling
    - **Semi-artificial** data: real-world data + synthetic labels



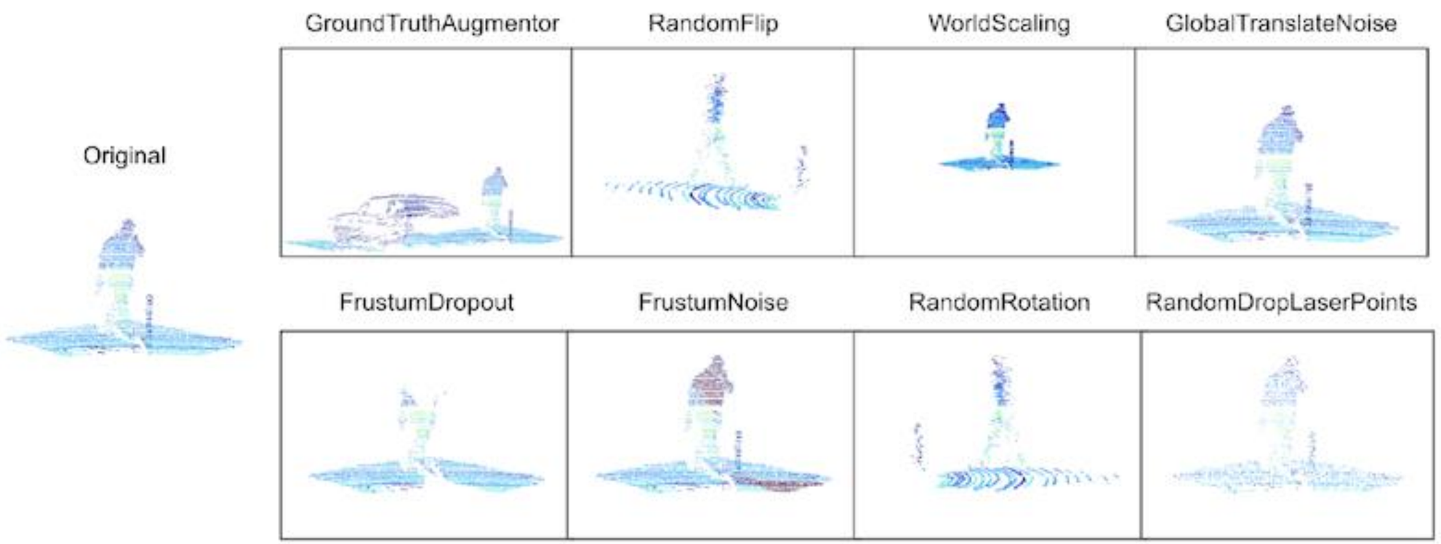
Images: ika, blogspot, unrealengine

Object Detection - Training Pipeline

Data Preprocessing

Preprocessing

- Dataset split in training / validation / testing
- Intensity normalization
- **Augmentation**
  - Flipping
  - Scaling
  - Cropping
  - Translating
  - Adding noise
  - Rotating



Images: medium

→ All steps can be applied to the global scene or individual objects

Object Detection - Training Pipeline

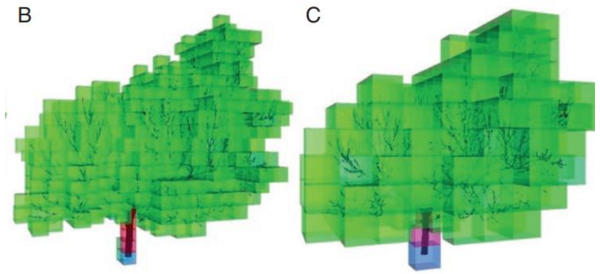
Network Architecture

- Input is **unstructured representation**
  - List of 3D coordinates
  - **Unknown** or variable point **amount**
  - Not always one **sensor source**

- **Range view**
  - Cylindrical projection
  - 2D image-like
  - Easy and efficient with CNN
  - Not feasible in fused data

- **Voxel based view**
  - Discretization along XYZ
  - Processing with CNN possible
  - Low runtime performace

Source: Triess et al. 2020



Source: Lecigne et al. 2018

Object Detection - Training Pipeline

Network Architecture

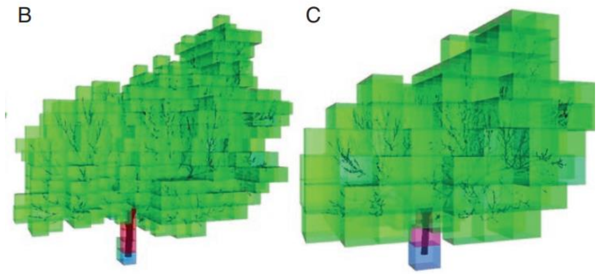
- Input is **unstructured representation**
  - List of 3D coordinates
  - **Unknown** or variable point **amount**
  - Not always one **sensor source**

- **Range view**
  - Cylindrical projection
  - 2D image-like
  - Easy and efficient with CNN
  - Not feasible in fused data

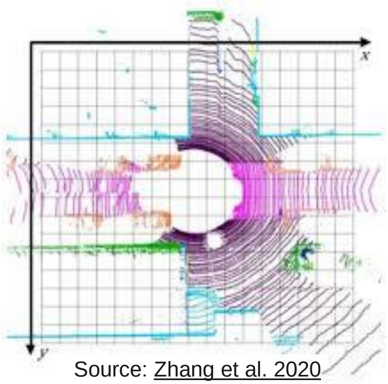
- **Voxel based view**
  - Discretization along XYZ
  - Processing with CNN possible
  - Low runtime performace

- **Bird eye view**
  - Structured grid in xy-plane
  - Max point amount in vertical pillars
  - z-axis encoded as features

Source: Triess et al. 2020



Source: Lecigne et al. 2018

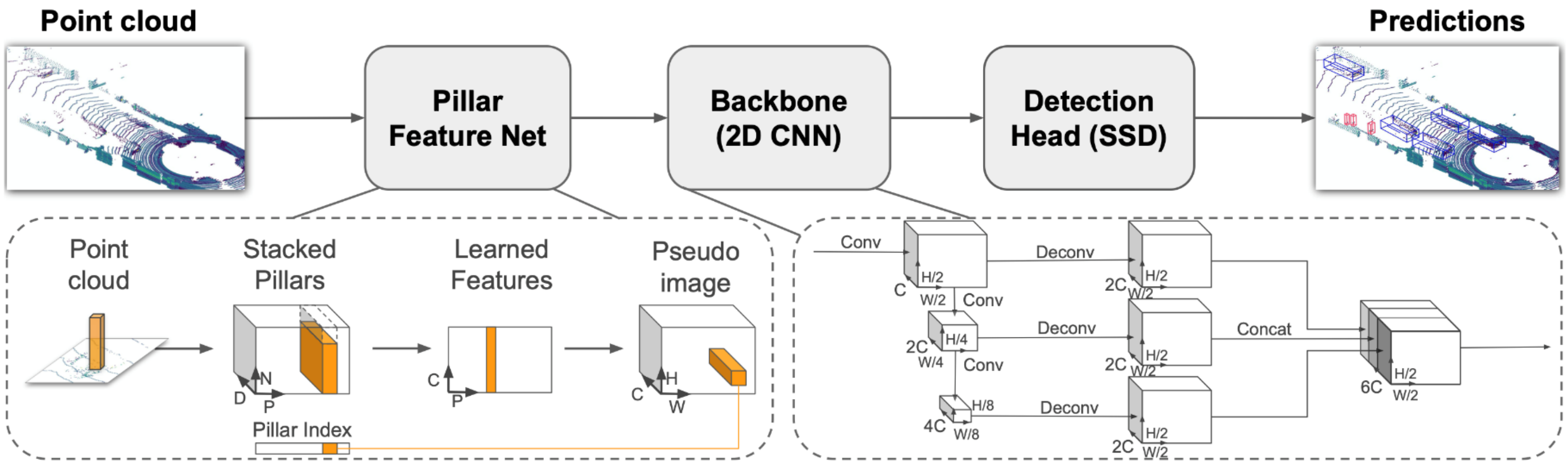


Source: Zhang et al. 2020

Object Detection - Training Pipeline

PointPillars

- Idea: efficient **feature encoding in vertical pillars** directly from raw pointclouds (**62 Hz** on KITTI - GPU)



Source: Lang et al. 2019

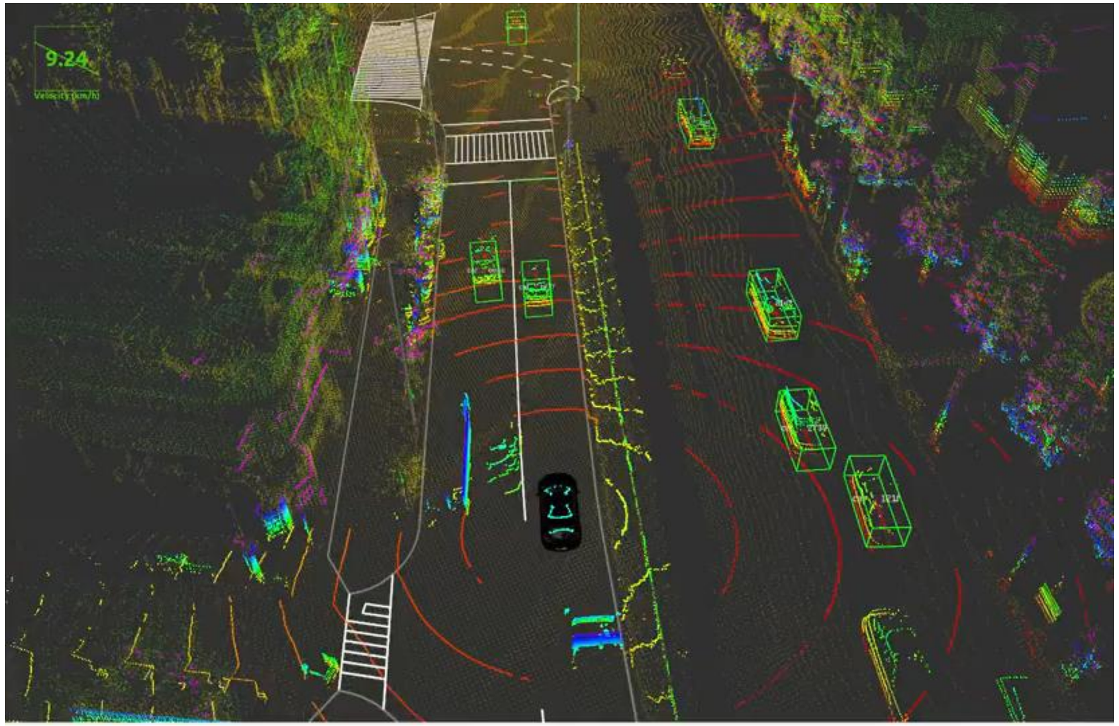
- Preprocessing: **create pillars** grid and anchors $P \times N \times D = 12000 \times 100 \times 9$
- Pillar Feature Net: extract **64 learnable features** for each grid cell
- Backbone: **2D CNN** with stacked grid cell features as input
- Detection Head: SSD for **prediction of target vector*** for each anchor and grid cell
- Postprocessing: **bounding box generation** based on target vectors and anchor boxes

*target vector contains: location, dimension, orientation, classification

Object Detection - Training Pipeline

Training

- Hyper **parameters**
  - batch size
  - epochs
  - learning rate
  - ...
  - anchor box setup
- **Training** (PointPillars)
  - 7481 KITTI training / validation samples
  - 160 epochs
- Inference
  - **88.7% AP** on *KITTI Car medium* benchmark
  - **62 FPS** on KITTI Benchmarks

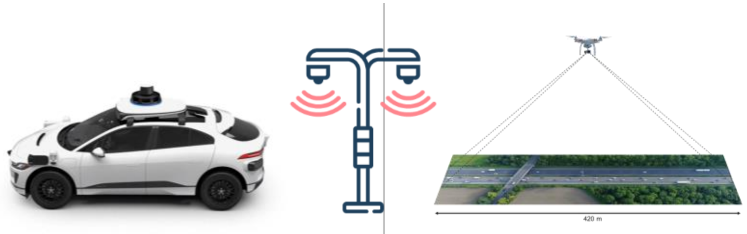


Video: youtube

Object Detection - Training Pipeline

Summary

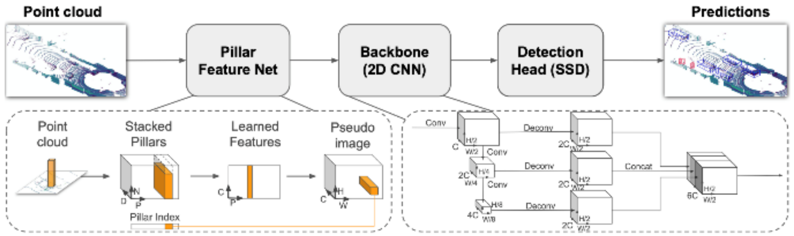
- Data **acquisition**



Images: ika, blogspot

- Data **preprocessing** (e.g. data split, normalization, augmentation)

- Different **approaches** for **3D** point clouds → **PointPillars** architecture



Source: Lang et al. 2019

- Training process with variety of **hyperparameters**