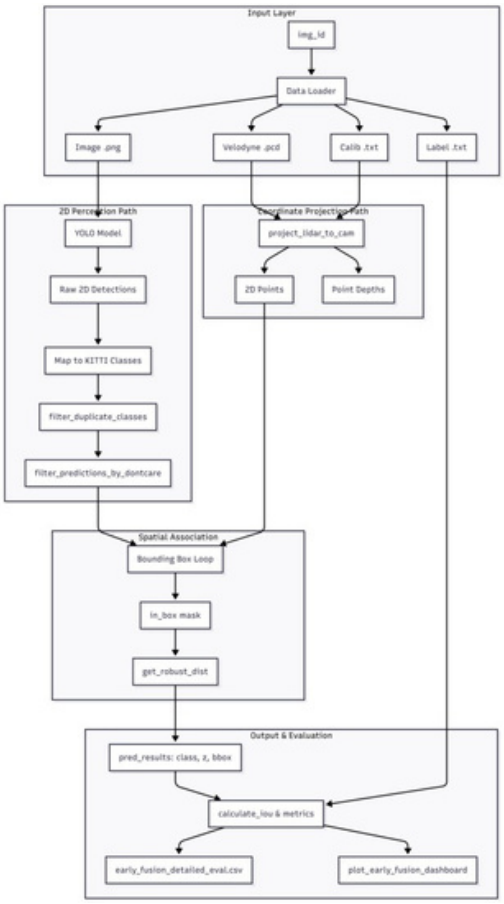
System Architecture Details & Analysis

Sensor fusion {Early & Late }FUSIONS

1. Executive Summary

This report details the development and evaluation of two distinct sensor fusion architectures—Early Fusion and Late Fusion—aimed at achieving high-precision 3D object localization. The system integrates a high-resolution 2D detector (YOLO 26V + SAHI) with a 3D LiDAR backbone (PointPillars). Key innovations include custom label arbitration logic for the KITTI dataset and the implementation of Slicing Aided Hyper Inference (SAHI) to overcome grid-based detection limitations.

2. System Architecture & Methodology

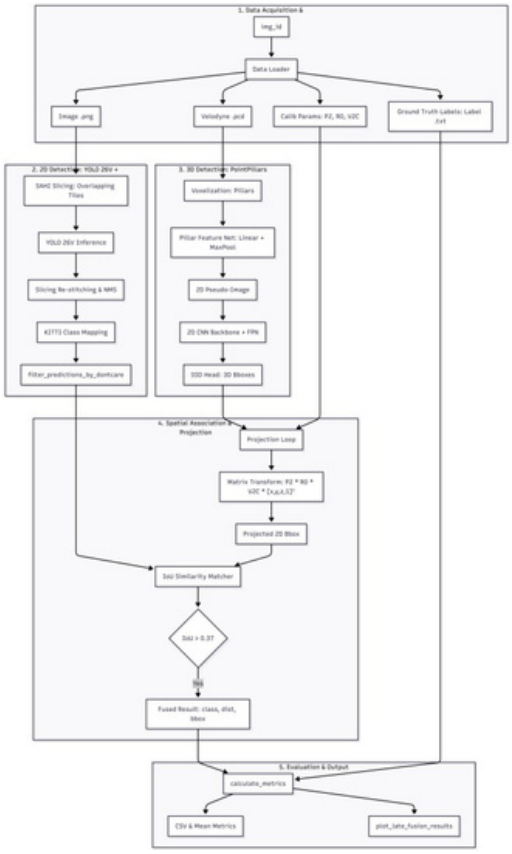


2.1. The 2D Pipeline: YOLO 26V + SAHI

Standard YOLO models utilize a fixed grid system for detection. While efficient, this often fails to capture fine-grained details in high-resolution images, especially for small or distant objects.

- The Problem: Small objects (Pedestrians/Cyclists) occupy too few grid cells to trigger reliable detections.
- The Solution (SAHI): I implemented Slicing Aided Hyper Inference. The image is divided into overlapping tiles (slices). Inference is run on each slice independently, preserving the original pixel density.
- Stitching Logic: To prevent duplicate detections at tile boundaries, I utilized Non-Maximum Merging. In cases where tiles produced overlapping boxes with conflicting classes, I implemented a Max-Score Arbitration rule—the label with the highest confidence score is retained.

2. System Architecture & Methodology



2.2. The 3D Pipeline: PointPillars Architecture
The 3D detection utilizes the PointPillars model, which I sourced from a GitHub implementation that leveraged Transfer Learning on the KITTI dataset.
Architecture Details:
Pillar Fold: Converts raw point clouds into vertical columns (pillars).
Pillar Feature Net (PFN): Learns a 9D feature vector for each point and uses a Max-Pool layer to create a sparse pseudo-image.
2D Backbone: A standard CNN processes the pseudo-image in Bird's Eye View (BEV).
SSD Head: Predicts 3D bounding boxes (x, y, z, w, l, h, θ).

3. Fusion Strategies

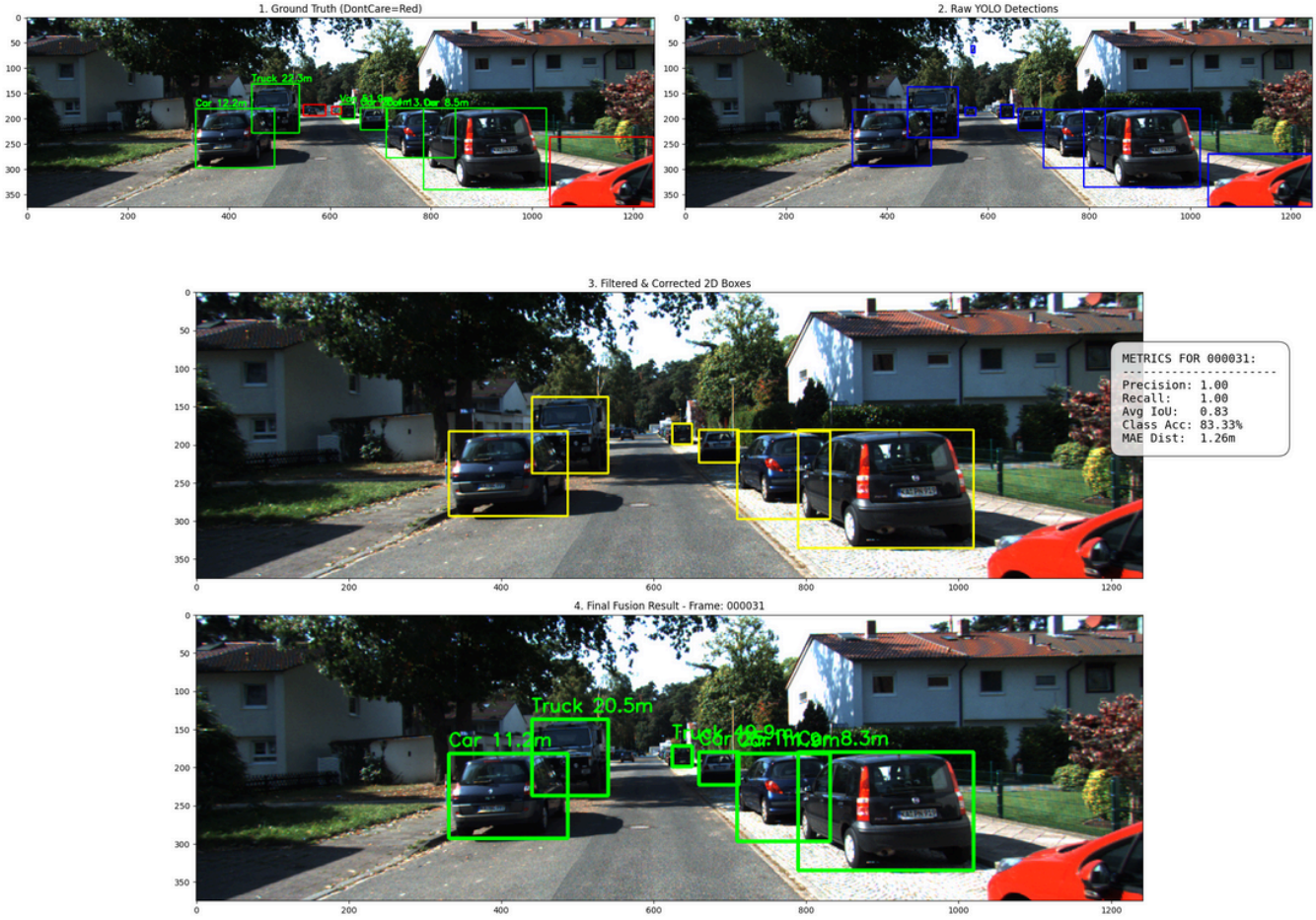
Step 1: Early Fusion (Data-Level)
In the early fusion script (early_fusion.py), raw LiDAR points are projected onto the 2D image plane using calibration matrices ($P_2 \cdot R_0 \cdot V_{2C}$).

- Process: 2D detections from YOLO 26V act as spatial filters. Only LiDAR points falling inside these 2D boxes are considered.
- Optimization: A "robust distance" algorithm calculates the median depth of points within the box to eliminate noise and edge-case outliers.

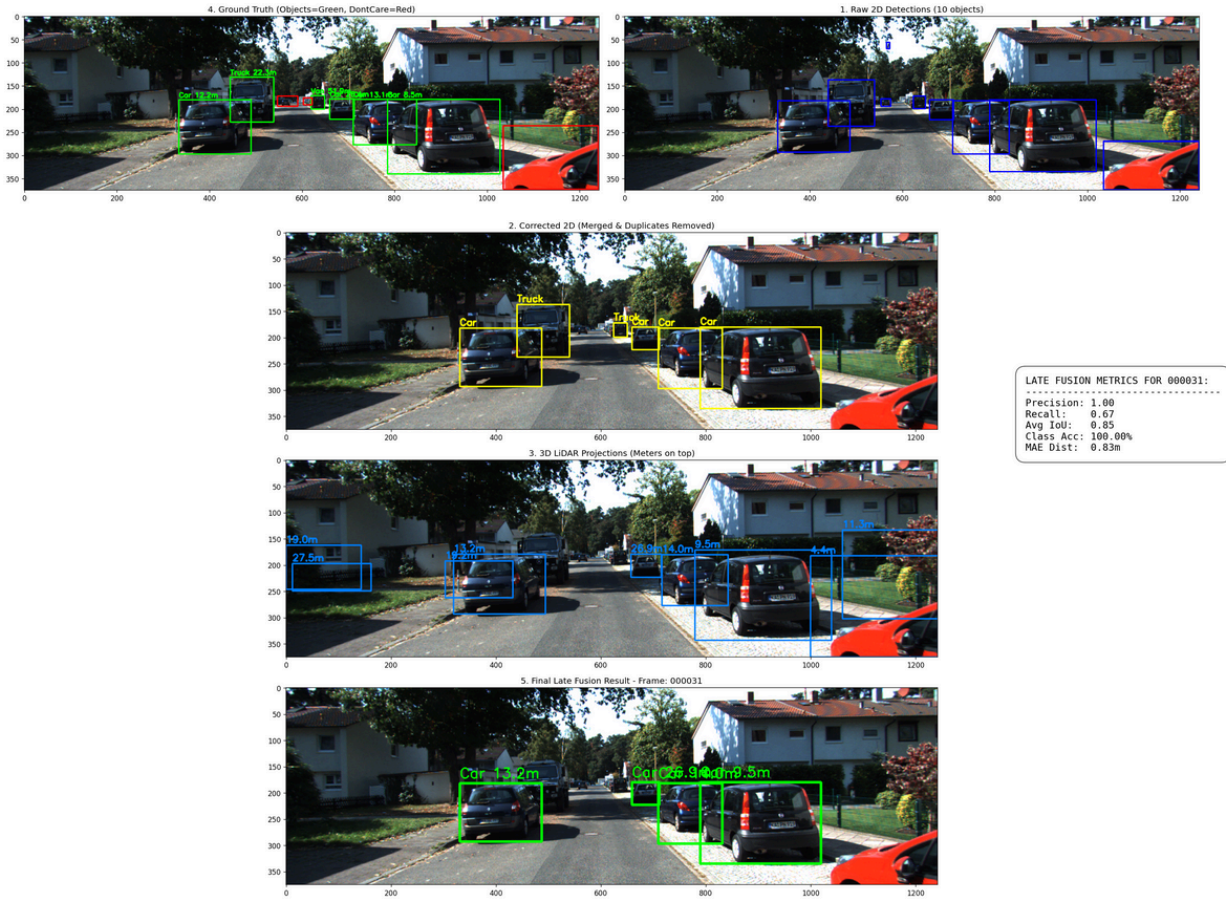
Step 2: Late Fusion (Object-Level)
In late_fusion.py, the outputs of the 2D and 3D models are generated independently and fused at the proposal level.

- Projection Logic: The 3D boxes from PointPillars are projected into 2D image coordinates.
- Association: We calculate the Intersection over Union (IoU) between the projected 3D box and the YOLO 26V 2D box. If IoU > 0.3, the objects are fused.

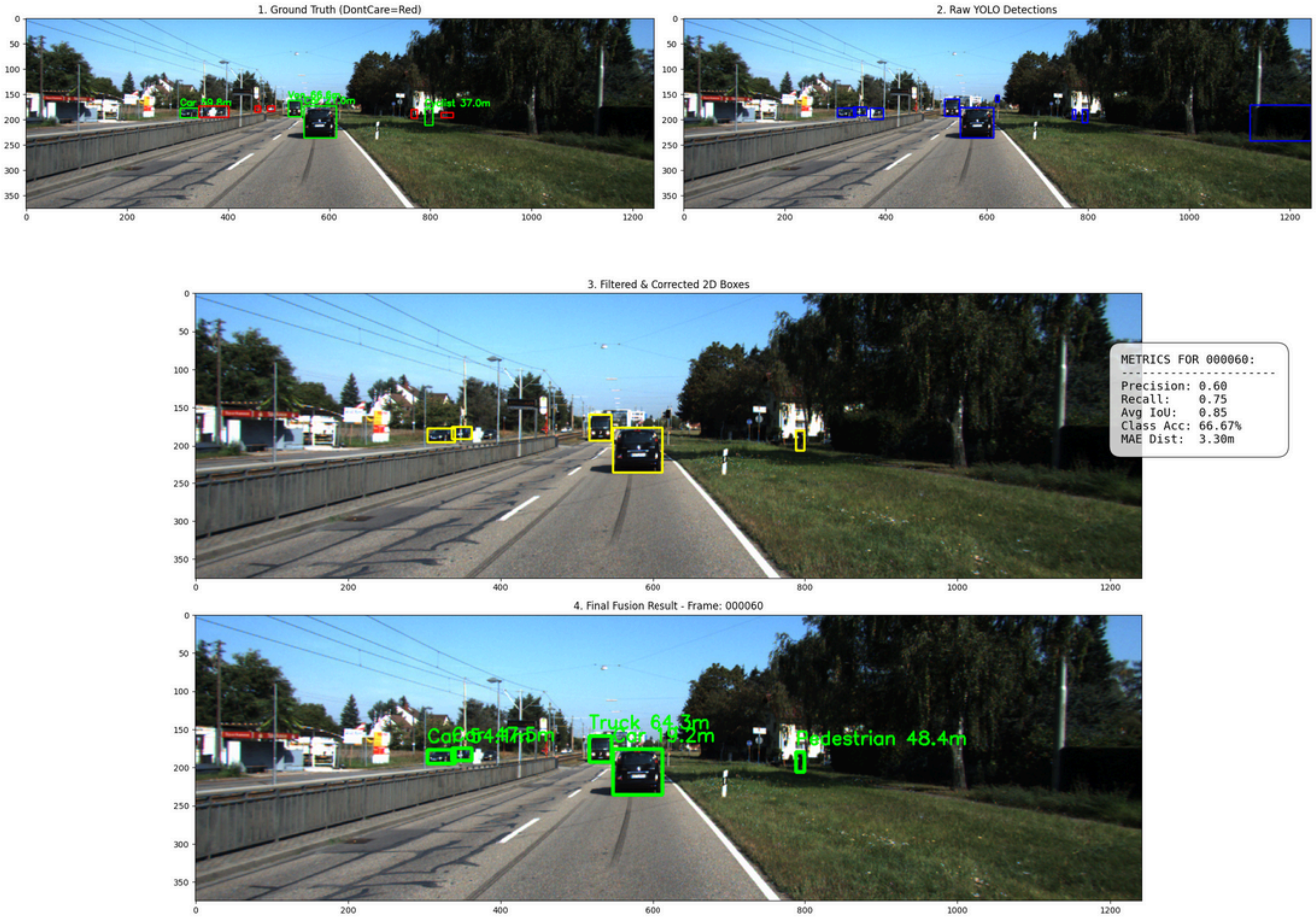
Sample Early fusion VS LATE fusion results:



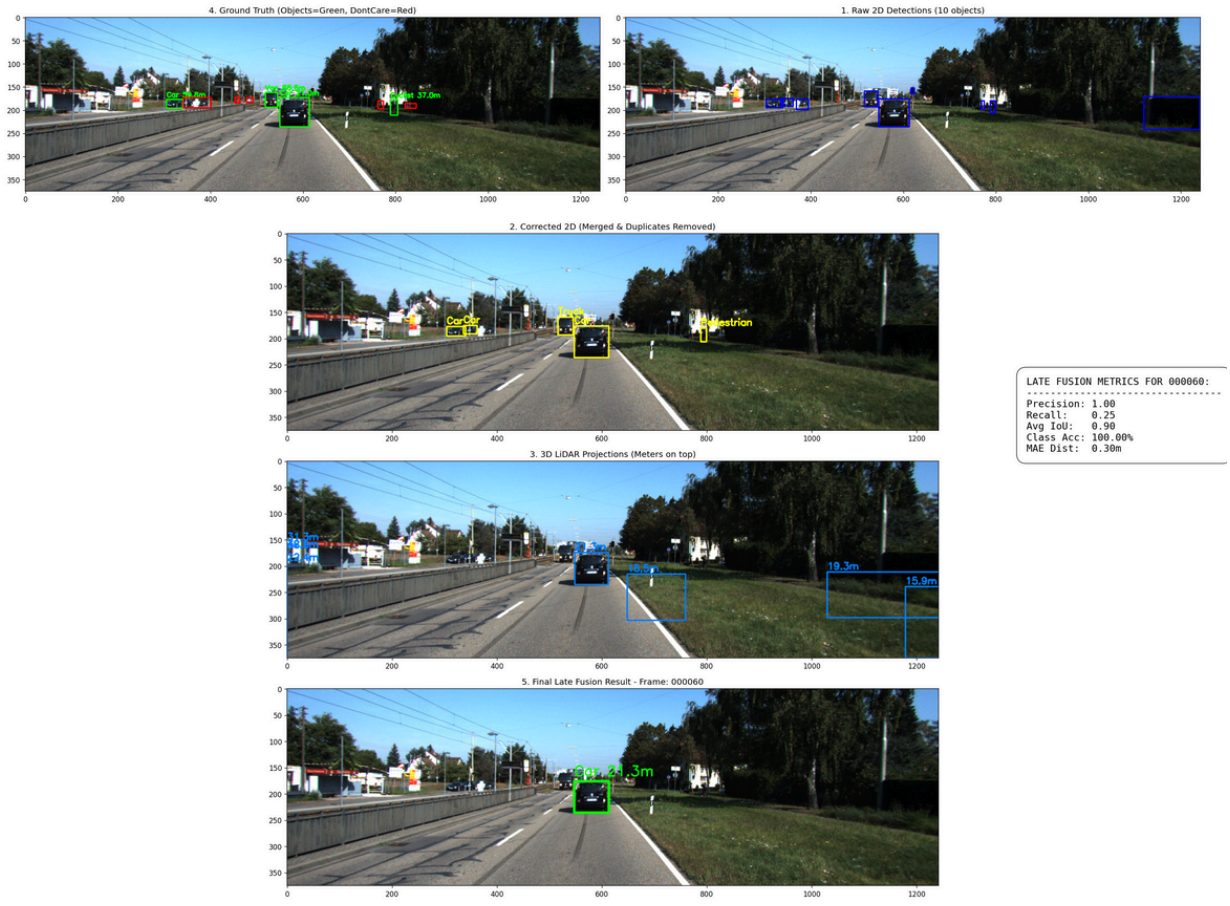
earlyfusion 00031 image



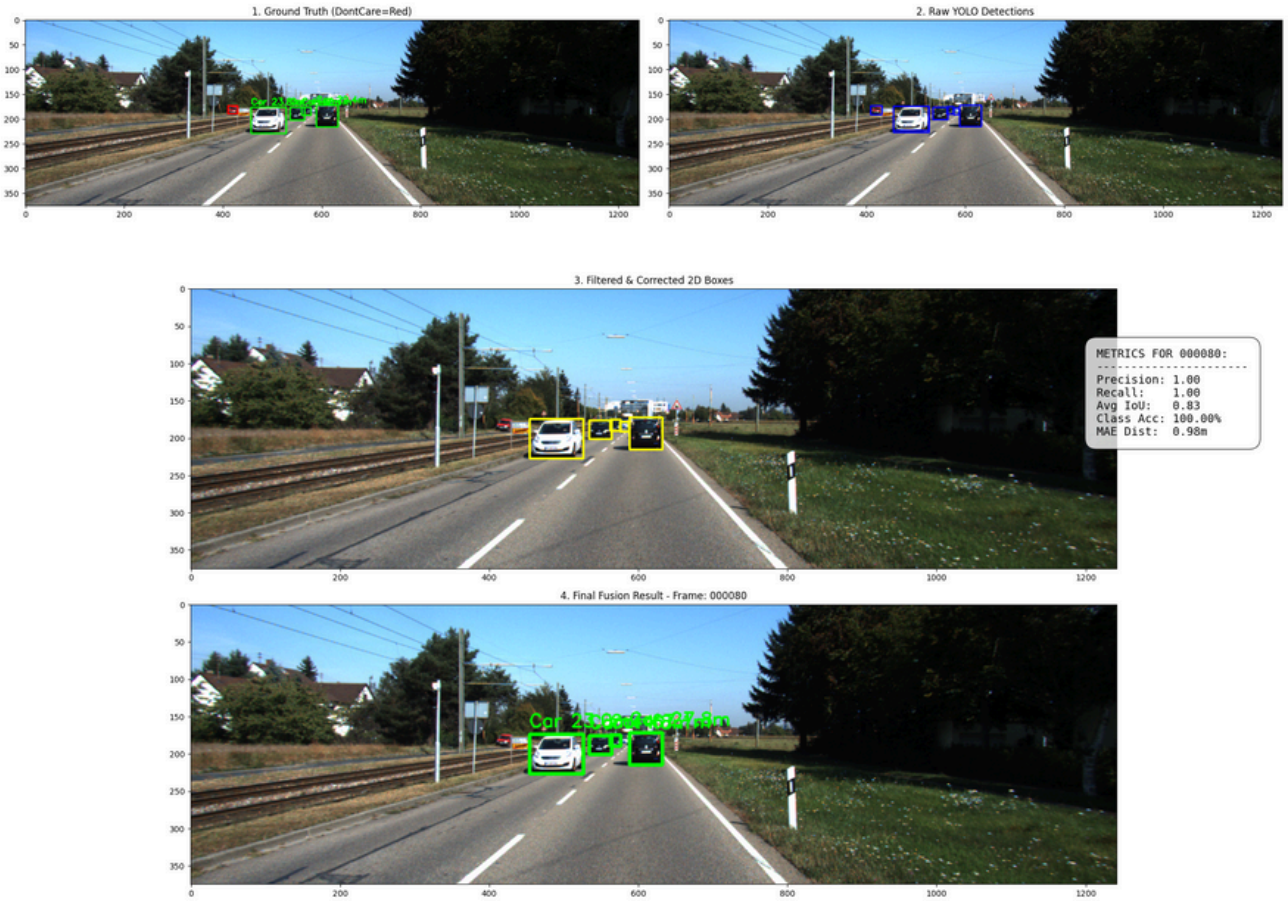
late fusion 00031 image



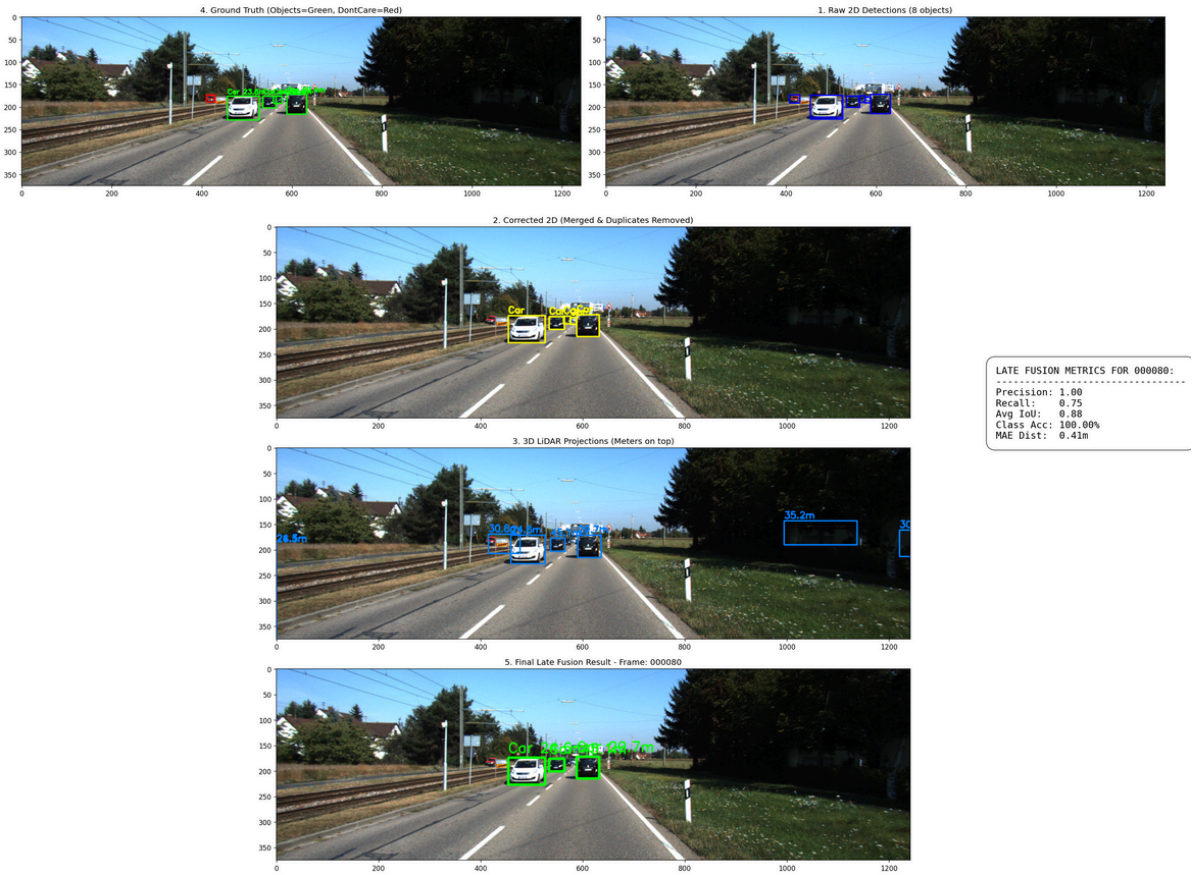
earlyfusion 00060 image



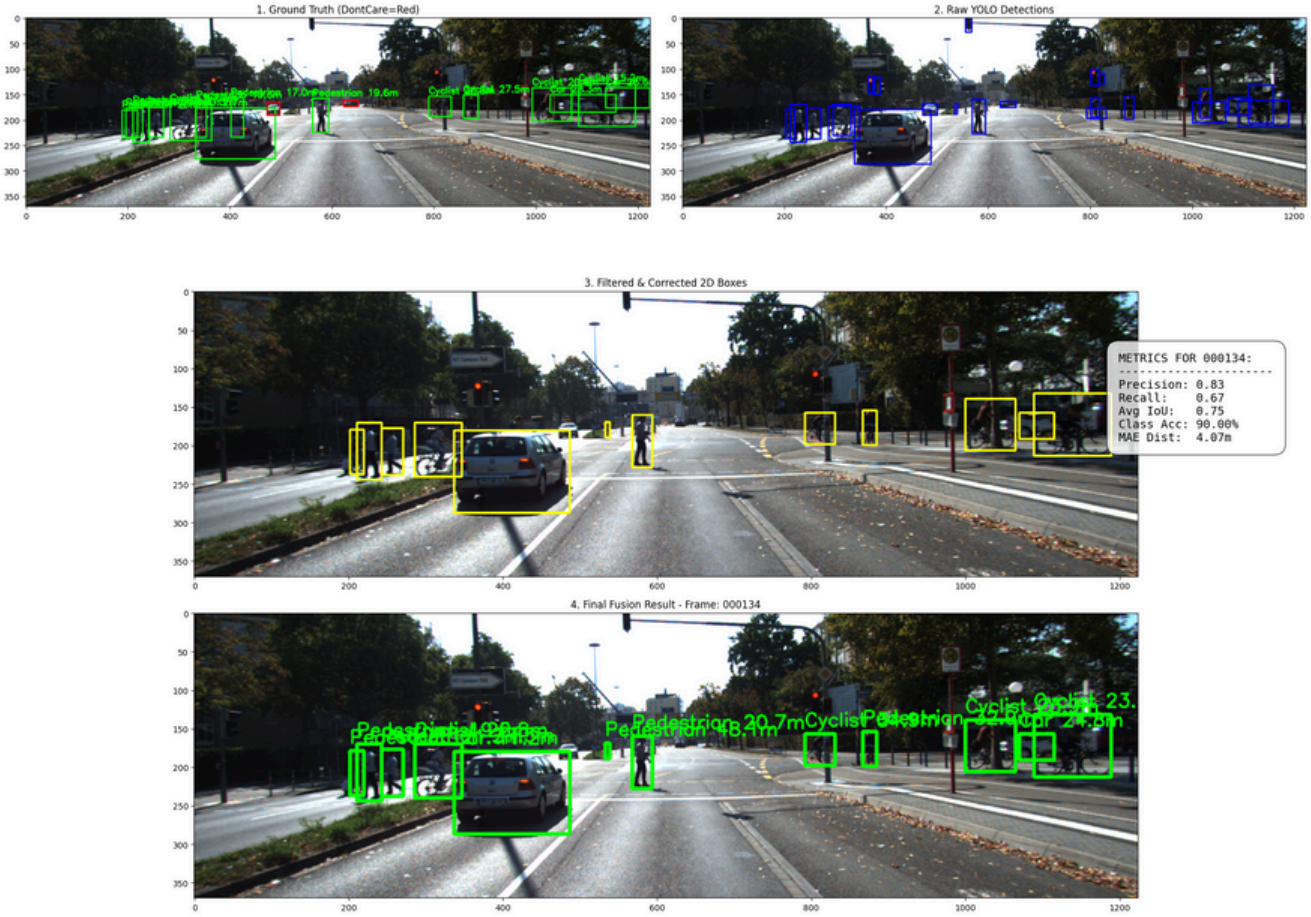
late fusion 00060 image



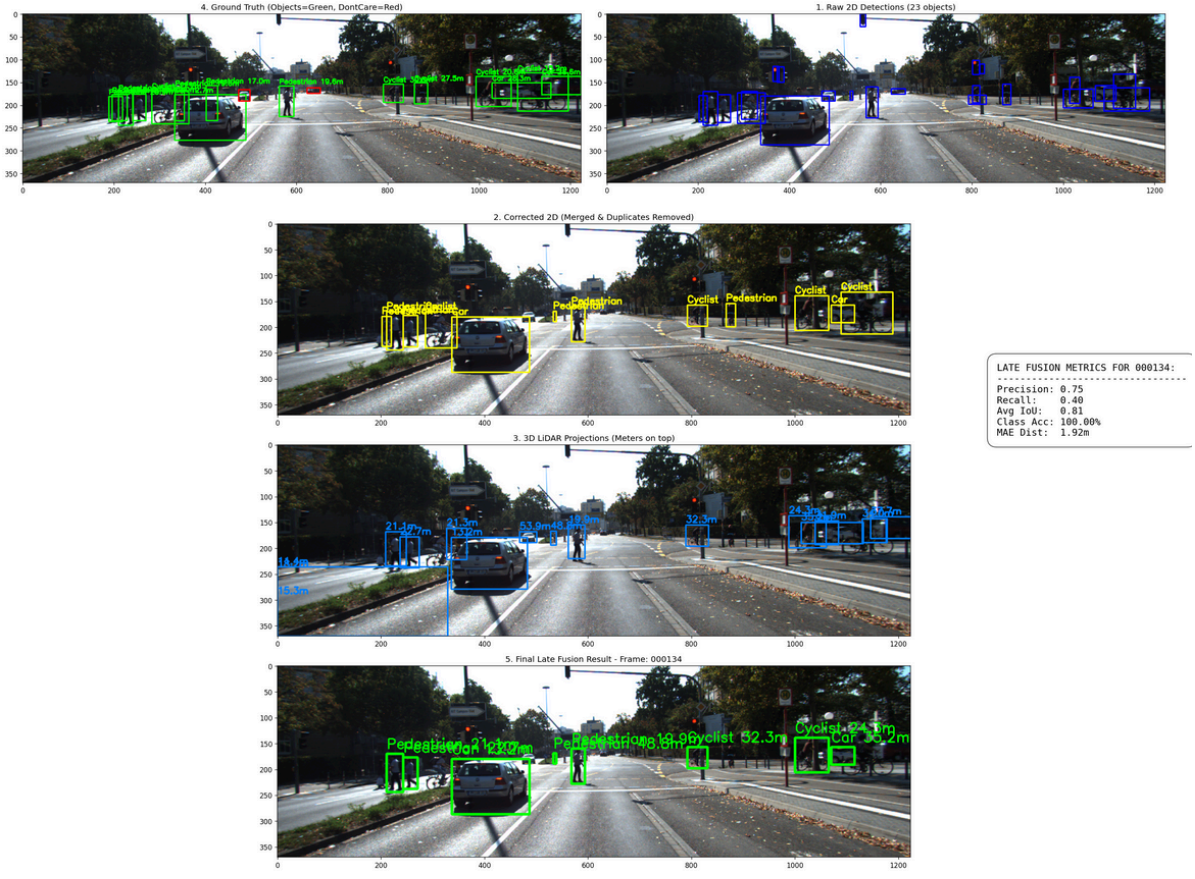
earlyfusion 00080 image



late fusion 00080 image



earlyfusion 00134 image



late fusion 00134 image

4. Challenges & Label Optimization

4.1. KITTI Label Misalignment

Standard YOLO pre-trained classes do not perfectly align with KITTI requirements. I developed a custom mapping layer to resolve this:

Cyclist Logic: If a Person detection overlaps with a Bicycle detection, the system merges them into a single Cyclist class.

Pedestrian Logic: Isolated Person detections are remapped to the Pedestrian class**.**

4.2. "DontCare" Region Handling

Initially, the evaluation produced misleadingly low accuracy. Upon analysis, I discovered the KITTI dataset contains "DontCare" labels (objects that are too small or occluded for human labeling).

Fix: I modified the evaluation loop to filter out any predictions falling within "DontCare" regions. This resulted in a significantly more accurate representation of the model's true performance.

5. Performance Analysis & Comparison

Metric	Early Fusion	Late Fusion
precision	0.847	0.91
recall	0.843	0.573
mIoU	0.825	0.861
class_acc	0.88	1.0
mae_dist	2.229	1.422

Note: Both models currently share a limitation regarding the "Van" class. Because neither the 2D nor 3D pre-trained weights included "Van" in their initial training set, they are often misclassified as "Car" or "Truck".

6. Conclusion & Future Work

The system successfully bridges the gap between 2D high-resolution imagery and 3D spatial data. The use of SAHI was pivotal in capturing small objects, while the custom arbitration logic ensured KITTI compliance.

Next Steps:

Retraining: Fine-tune both YOLO 26V and PointPillars specifically on the KITTI "Van" and "Tram" classes to resolve the current recall limitations.

Temporal Fusion: Implement Kalman Filtering to track objects across frames, reducing "flicker" in detections.