InfraDet3D: Multi-Modal 3D Object Detection based on Roadside Infrastructure Camera and LiDAR Sensors

Walter Zimmer [ID] ,   Joseph Birkner [ID] ,   Marcel Brucker [ID] ,   Huu Tung Nguyen [ID] ,

Stefan Petrovski [ID] ,   Bohan Wang [ID] ,   Alois C. Knoll [ID]

Abstract—Current multi-modal object detection approaches focus on the vehicle domain and are limited in the perception range and the processing capabilities. Roadside sensor units (RSUs) introduce a new domain for perception systems and leverage altitude to observe traffic. Cameras and LiDARs mounted on gantry bridges increase the perception range and produce a full digital twin of the traffic. In this work, we introduce *InfraDet3D*, a multi-modal 3D object detector for roadside infrastructure sensors. We fuse two LiDARs using early fusion and further incorporate detections from monocular cameras to increase the robustness and to detect small objects. Our monocular 3D detection module uses HD maps to ground object yaw hypotheses, improving the final perception results. The perception framework is deployed on a real-world intersection that is part of the *A9 Test Stretch* in Munich, Germany. We perform several ablation studies and experiments and show that fusing two LiDARs with two cameras leads to an improvement of $+1.90$ `mAP` compared to a camera-only solution. We evaluate our results on the A9 infrastructure dataset and achieve 68.48 `mAP` on the test set. The dataset and code will be available at **https://a9-dataset.com** to allow the research community to further improve the perception results and make autonomous driving safer.

Index Terms—3D Perception, Camera-LiDAR Fusion, Roadside Sensors, Infrastructure Sensors, Autonomous Driving

I. INTRODUCTION

Roadside perception is vital to improve the situation awareness and to provide a far-reaching view for automated vehicles. Roadside sensors installed on infrastructure systems like the A9 Test Stretch [2], [3] increase the perception range drastically. They perceive objects around the corner, e.g. to warn drivers performing a left or right turn. A cost-effective solution is needed to process perception models in real-time and provide accurate results at the same time.

Positional data captured from roadside sensors is sent through high performance units to all traffic participants to decrease blind spots and prevent accidents. It has been shown that roadside sensors increase the situation awareness by sending important notifications and warnings to vulnerable road users (VRUs) and drivers [4]–[6]. In this work, we contribute to the challenge of sparse point clouds in the domain of roadside perception in the following way:

---

[1]The authors are with the School of Computation, Information and Technology (CIT), Department of Informatics, Technical University of Munich, TUM, 85748 Garching-Hochbrueck, Germany.
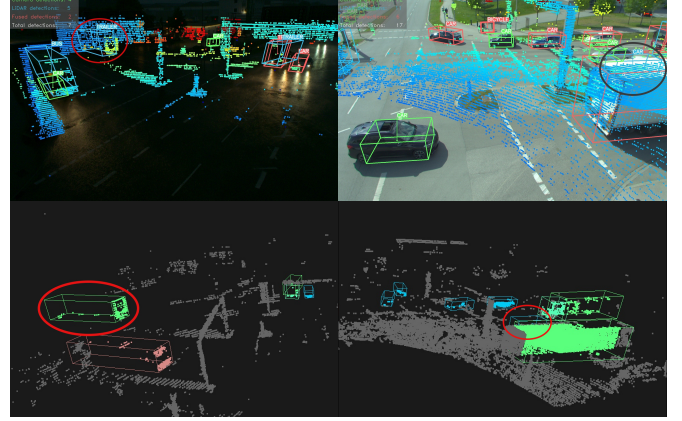Contact: `walter.zimmer@tum.de`

Fig. 1: Early and late fusion of two roadside cameras and LiDARs. We register point clouds from two LiDARs using G-ICP [1] and project them with the camera-LiDAR detections into the image. *Left column:* Night detection results in more and better classified LiDAR detections. *Right column:* Detections during day time demonstrate a 41.67% increase in detections using the fusion approach. Moreover, even occluded objects, like the car behind the trailer (right) or the truck behind the gantry bridge (left), can be detected with our *InfraDet3D* Fusion Framework.

- We propose a real-time point cloud registration algorithm to register infrastructure LiDARs which enhances the point density. Our experiments show that early fusion of point clouds leads to an increase of $+1.32$ `mAP`.
- Fusing supervised and unsupervised LiDAR 3D object detectors increases the robustness and reduces the number of false positive detections.
- We connect our perception module to real HD maps (+2.7 `mAP`) of the *A9 Testbed* to extract road information, as well as to validate and filter the perception results.
- Our camera-LiDAR fusion module further enhances the robustness of our whole perception toolbox (+1.62 `mAP`) by providing perception results during day and night time.
- Finally, we evaluate all 3D detectors on the A9-I dataset and introduce a leaderboard to allow the research community to benchmark their models on our dataset.
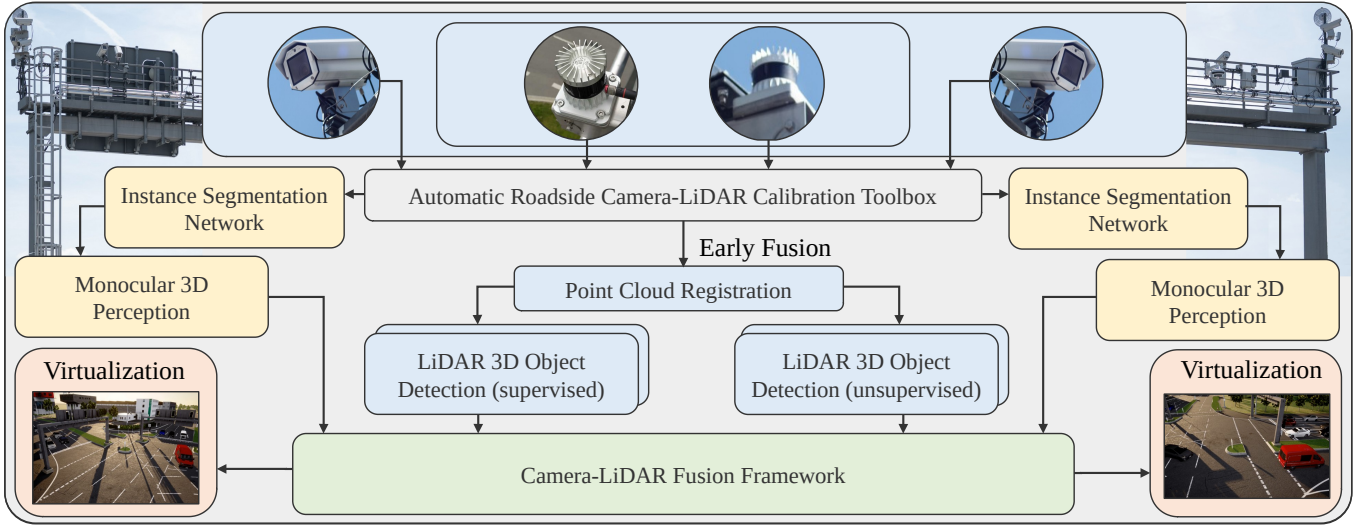
Fig. 2: *InfraDet3D* Perception Framework Architecture. Our proposed model is deployed on a real intersection (S110) part of the A9 Test Stretch for Autonomous Driving in Munich, Germany.

II. RELATED WORK

Much research has been done in the area of roadside 3D perception. Traditional approaches [7] increase the robustness of roadside LiDAR perception systems because of the similarity and the lack of diversity in the background point cloud. Furthermore, they do not require labeled data and process point clouds efficiently. In [8] a 3D vehicle detection approach is proposed that uses a single camera. First, they segment the instance mask in the image, extract the bottom contour and project it on the road plane to get the 3D position. Then, they cluster the projected points into objects by applying K-means clustering. Afterwards, they estimate the dimensions (length and width) and orientation (heading angle) of vehicles by fitting a box for each cluster. Finally, they refine the 3D box to fit it within the 2D box by maximizing the posterior probability. Bai et al. proposes a learning-based approach [9] that requires huge labeled datasets and performs poorly in domains where no labeled data is available. The authors introduce a real-time LiDAR-based traffic surveillance system to detect objects in 3D. They develop *3DSORT*, a 3D multi-object tracker by extending *DeepSORT* [10]. The limitation of all mentioned approaches is that they have no labeled training data of roadside LiDARs and use open-source datasets like *nuScenes* [11] to train the model. To the best of our knowledge there is no roadside 3D perception framework available that is able to fuse data from multiple road side sensor units. Furthermore, there is no solution that combines different fusion levels (early and late fusion), as well as traditional and learning-based approaches into a single framework.

III. A9 INTERSECTION DATASET

The A9 Intersection (A9-I) dataset is an extension of the A9 Dataset [12]. It contains labeled data (in *OpenLABEL* format) of two cameras and two LiDAR sensors mounted on the S110 gantry bridge that is part of the *A9 Test Stretch for Autonomous Driving*. It contains 9,600 labeled point clouds and images with 57,743 labeled 3D objects ($\varnothing$12/frame) and is split into a training (80%), validation (10%), and test set (10%). The test set contains a sequence with labeled track IDs and sampled frames from four different scenarios. We applied stratified sampling to balance the dataset among sensor types and scenarios. The set contains 25% night data with severe weather conditions like heavy rain which allows the model to perform well under challenging weather conditions. Our dataset was created by labeling experts and some improvements were done to further enhance the label quality using the *proAnno* labeling toolbox which is based on [13].

IV. SENSOR CALIBRATION

In our framework multiple roadside LiDAR and camera sensors are fused and processed together for the detection task. Our automatic calibration of infrastructure LiDARs and cameras, which outputs the precise pose of these sensors, is the most fundamental part of the framework. In order to calibrate the sensors in the real world, we propose an automatic target-less LiDAR-camera calibration model. We use the calibration method proposed in [14] as a baseline and extend it to outdoor scenes captured by infrastructure roadside sensors of a different manufacturer. To improve the robustness of the model under different external conditions, such as different scene complexities, lighting conditions, or sensor conditions, we introduce various automatic preprocessing submodules (see Figure 3).

First, we undistort the input images. After that, an automatic background cropping (based on monocular depth estimation [15]) is employed to remove the background objects. If there is shadow on the ground, the automatic shadow filtering module will be activated to filter the shadow. After the preprocessing, the *Canny* edge detector [16] is adopted to extract 2D edges
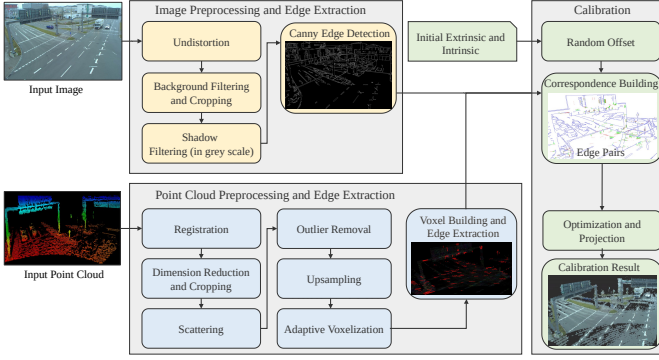
Fig. 3: Automatic calibration pipeline. We integrate four camera image and seven LiDAR point cloud preprocessing modules into our pipeline in order to increase the robustness of real-world outdoor calibration of roadside sensors. The algorithm takes the image and point cloud that is published continuously on the live system as input and outputs both, the calibration results and qualitative projections of point clouds into camera images.

in images.

For LiDAR preprocessing, point clouds from three LiDARs are registered to the target LiDAR. The input point cloud is cropped and only four dimensions are preserved (x,y,z and intensity). Scattering is applied to increase the density of single frame point cloud scans. Afterwards, the point cloud is automatically subdivided into ground and non-ground point clouds. Outlier removal is applied to the ground point cloud to filter the noise in order to preserve more points of the gantry bridge. We also use point upsampling [17] to improve the surface texture of point clouds. After the preprocessing, voxels are extracted from the point clouds. For faster extraction, adaptive voxelization [18] is introduced. *RANSAC* plane fitting is applied to extract planes within the voxel. The intersections among planes are extracted as LiDAR edge clouds.

After the edges are extracted from the point cloud, they are projected into the image and correspondences between LiDAR and camera edges are established. A cost based on maximum likelihood estimate is optimized and the qualitative result is generated. Our automatic calibration model demonstrates good robustness against different weather conditions and traffic scenarios in the intersection and provides accurate extrinsic calibration values for the perception framework.

V. MONOCULAR 3D OBJECT DETECTION

Due to their low cost and high output information density, monocular RGB cameras are incorporated as sensors into the *InfraDet3D* architecture. The monocular detection pipeline is based on an augmented *L-Shape-Fitting* algorithm as proposed by [19]. The basic *L-Shape-Fitting* algorithm has also been used in other recent roadside infrastructure perception architectures, such as the detector for the MONA dataset [20] and the Cooperative Vehicle Infrastructure System 3D detector [8]. However, the augmentation of this algorithm with

object tracking, to score yaw hypotheses based on historical plausibility, is novel. Furthermore, we propose the integration of the High-Definition (HD) map to limit yaw hypotheses with regard to matching lanes. Both features are inspired by TrafficNet [21] and UrbanNet [22] architectures. An overview of the full monocular detection pipeline is given in Figure 4.

A. From 2D Instance Masks to 3D Bottom Contours

We use the YOLOv7 Instance Segmentation model [23] on RGB camera frames. The RGB frames are downscaled to 1280x720 pixels size, to accelerate the instance segmentation runtime. The instance masks, which are output from the model, are processed to extract the bottom image contour from the masks. The 2D bottom contour coordinates for each mask are then projected from screen-space to 3D intersection space via raycasting. Finally, the *DBSCAN* (Density-Based Spatial Clustering of Applications with Noise) [24] algorithm is applied to denoise each detection's 3D bottom outline.

B. HD Map Yaw Candidate Lookup

Using lane geometry from an HD map of the sensor-covered areas, each lane's road surface is rasterized into a heading lookup grid, covering the field of view of the respective camera. The heading lookup grids are rendered to a resolution of 10x10 cm grid cells. Each grid cell C_{ij} is a set $\{(\text{lane\_id}_k, \theta_k)\}_{k=0}^{k \leq N_{ij}}$ of lane ID and heading pairs which apply to the respective cell. The heading for a lane at the position of the grid cell is interpolated from the direction of the surrounding lane borders. At inference time, for each 3D bottom contour point of a detected object, the grids are queried to compute a set $L = \{(\text{lane\_id}_i, \theta_i)\}_{i=0}^{i \leq N}$ of possible heading values along the bottom contour. This set is aggregated into a histogram with hit counts and average heading angle per lane ID. The hit counts for each lane ID are normalized into confidence values in the range of $[0, 1]$ through division over the maximum hit count value. This yields one $H_j = \{(\text{lane\_id}_i, \theta_i, \text{confidence}_i)\}_{i=0}^{i < M}$ three-tuple-set of possible heading values for each instance j.

C. Augmented L-Shape-Fitting

The *L-Shape-Fitting* (LSF) algorithm searches for a rectangle that fits a specific bottom contour by maximizing a score value[1], which is calculated as a function of a rectangle yaw (θ) hypothesis and the 3D bottom contour points. In the basic form, the algorithm simply goes through several θ values from the range $[0, \pi]$ at fixed increments. In our augmented version of the algorithm, we only run LSF for θ values as present in the HD map lookup histogram for each 3D bottom contour. Furthermore, we multiply the calculated score value for each yaw hypothesis with the respective confidence value from the normalized map lookup histogram. Finally, the score is also multiplied with a historical plausibility factor. The calculation of this factor is explained in the following.

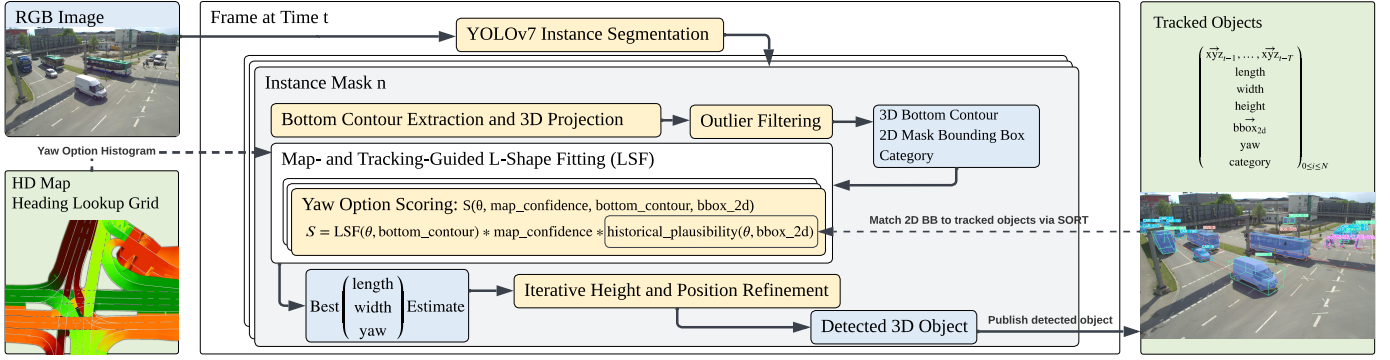
[1]Such as negative average variance

Fig. 4: Monocular 3D object detection pipeline, grounding shape hypotheses via tracking and the HD map.

Using a screen-space SORT tracker [25], we match a detected object's bounding box to detections from previous frames. For a successfully matched detection, historical 3D position values $L = \{\vec{l}_{t-1}, \ldots, \vec{l}_{t-T}\}$ are retrieved. Given the historical positions L and a position hypothesis $\vec{l}_t(\theta_t)$, the historical plausibility score HP for a yaw hypothesis θ_t is calculated as in the following equation:

$$\text{HP} = \prod_{\delta_t=1}^{\delta_t \leq T} \pi/2 - \Delta_{\angle}(|\theta_t - \text{atan2}(\vec{l}_t(\theta_t) - \vec{l}_{t-\delta_t})| \bmod \pi)$$

The Delta-Angle function $\Delta_{\angle} : [0, \pi) \to [0, \pi/2)$ converts the passed raw angular difference, which is already less than π, into a value less than $\pi/2$ by returning angular deltas $\delta_{>\pi/2}$ larger than $\pi/2$ as $\pi - \delta_{>\pi/2}$. This ensures that a yaw hypothesis, which is parallel, yet opposed to a historical orientation, is not erroneously punished. In practice, we have implemented a threshold of six historical positions that are evaluated to determine the plausibility of a yaw hypothesis.

D. Height Estimation and Dimension Filtering

The height for each detection is initialized from a fixed value for the object type of the detection. Both the height and the location are then jointly optimized through binary search, until the estimated projected 2D object height and the original mask height are the same by $\epsilon < 1\text{px}$. The length and width values, as estimated by the *L-Shape-Fitting* algorithm for each 3D bottom contour, are limited to minimum and maximum values, which are also looked up per object category.

VI. LiDAR 3D Object Detection

A. Unsupervised 3D Object Detection

LiDAR sensors are a popular choice for roadside object detection as they provide accurate 3D information in a large field of view and are lighting invariant. Studies on roadside LiDAR object detection favor traditional approaches based on clustering. Before clustering an extracted foreground point cloud into individual objects, these studies discard the ground, walls, trees, and other background artifacts from the raw point cloud. To discard the irrelevant background, our first 3D LiDAR object detector uses a fast four step procedure. First, the detector crops a predefined region of interest, which

always remains the same as the LiDAR sensor is installed statically on roadside infrastructure. This first step removes 69.9% of points on average. Second, the detector finds points belonging to the ground by considering the Euclidean distance to a predefined plane model together with a threshold of 0.2 m. Third, the detector filters background artifacts within the region of interest based on the coarse-fine triangle algorithm [26]. The fourth step is radius outlier removal ($n = 15$, $r = 0.8$), which refines the extraction of the foreground point cloud. The remaining foreground point cloud represents all traffic objects, including stationary ones. It is divided into distinct point clusters, each corresponding to a potential road user, by *DBSCAN* ($\epsilon = 0.8$, $n_{min} = 3$). Around each point cluster, the detector fits an oriented 3D bounding box using its convex hull and principal component analysis. Finally, the detector classifies the localized objects by means of object dimensions and point density.

B. Supervised 3D Object Detection

For the data-driven approach, we are using *PointPillars* [27] which runs with a fast inference rate of 38 FPS. In comparison to the unsupervised approach, we can input the registered point cloud (262k points) directly into the model, consisting of three modules. In the first step the *PillarFeatureNet* converts the point cloud into a sparse pseudo-image. After obtaining the pseudo-image, the 2D backbone produces features at a small spatial resolution. Theses features are then upsampled and concatenated. In the last step, an anchor-based detection head tries to match the bounding boxes to the ground truth.

We used the *PointPillars* implementation of *OpenPCDet* [28] and adapted it to our A9 intersection dataset. For training, we limited the point cloud range from -64 to 64 m in x-y direction and from -8 to 0 m in z direction. In the feature extraction step, we set the voxel size to $[0.16, 0.16, 0.8]$. The model was trained on 10 classes for 160 epochs and optimized using Adam with a learning rate of $\alpha = 0.003$, weight decay of 0.01 and cyclic momentum of $\beta = 0.9$.

VII. Multi-Modal 3D Object Detection

For the fusion of both modalities (LiDAR and camera detections) a late fusion technique is applied (see Fig. 5).
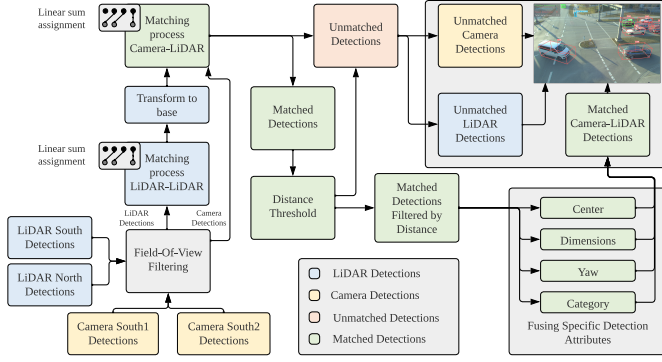
Fig. 5: Multi-modal 3D object detection pipeline. We apply a camera field-of-view filtering for all detections.

A. Data Association

A widely adopted method for combining and matching sensor data at the later stage is through data association, also defined as the linear assignment problem (LAP). It finds a one-to-one mapping between two sets of elements, such that the sum of the assigned pairwise costs is minimized.

The *Jonker-Volgenant* algorithm [29] is a method for solving the LAP and is based on augmenting paths. The algorithm starts by finding an initial feasible solution, e.g. by using the *Hungarian* algorithm [30]. Then, it repeatedly searches for an augmenting path, a path of alternating unmatched and matched elements that starts and ends at an unmatched element, and increases the number of assigned elements by one. The algorithm stops when no augmenting path can be found - the solution is optimal.

The *modified Jonker-Volgenant* algorithm [31] is a variation of the original one that improves its performance by using a heuristic search strategy. The heuristic builds on the idea of prioritizing the search for augmenting paths that are expected to have a high gain in terms of reducing the total cost.

In this work, the *modified Jonker-Volgenant* algorithm is chosen due to its increased speed ($O(n^3)$ [31]) in comparison to its variants. It also works well with non-integer costs. In our case, the matching process took $0.008\ ms$ on average per frame on the test set on a AMD Ryzen 5800X 8-Core CPU with an average number of 14.55 objects per frame.

B. Early Fusion of LiDAR Sensors

Our first fusion module combines multiple point cloud scans from different LiDAR sensors at time step *t* into a single dense point cloud. We preprocess the point clouds, as described in [5]. First, we downsample the point cloud and estimate point normals. Then, we compute a 33-dimensional FPFH[2] feature vector [32] for each point. This feature describes the local geometric property of each 3D point. Afterwards, we register several point clouds from roadside LiDARs that are time-synchronized with an NTP time server. The point cloud registration algorithm makes use of Fast Global Registration

---

[2]Fast point feature histogram

[1] to provide an initial transformation. For the refinement of the transformation, we use point-to-point ICP [33] as it leads to a lower RMSE value (0.448 m) than point-to-plane ICP. The full registration pipeline of two Ouster OS1-64 LiDARs takes 18.36 ms (54 FPS) on an Intel Core i7-9750H CPU and a voxel size of 2 m.

C. Late Fusion of LiDAR Sensors

For the LiDAR-to-LiDAR late fusion, we operate in LiDAR coordinate space. We transform the detections obtained by the unsupervised LiDAR detector and the supervised LiDAR detector into a common coordinate system. We match detections based on a distance of 3 m between their central positions. Matched detections are merged by selecting the central position and yaw vector of the detected object from the LiDAR sensor closest to the detection. Dimensions of the merged detections are computed by calculating the mean average of the detections from both detectors. Additionally, all unmatched detections are also included in the final result, resulting in an increase of 12.93% in the number of detections compared to using only a single LiDAR sensor.

D. Camera-LiDAR Late Fusion

For the camera-LiDAR fusion, we transform the LiDAR detections into the base coordinate system of the gantry bridge, which serves as the coordinate system for obtaining the monocular detections. This step is crucial for computing the inter-detection distances between camera and LiDAR instances based on their respective center positions. After the linear sum assignment, the matched detections are further filtered by a distance threshold of 3 m. The attributes of the matched detections are merged by eliminating matched camera detections and retaining only matched LiDAR detections, as they demonstrate greater accuracy on average during evaluation. The integration of the HD map leads to a substantial improvement (see Table III) in the camera yaw result, however it remains inferior to the results obtained from LiDAR. Table II displays the dependence of the mAP increase on the various attributes.

VIII. EVALUATION

A. Monocular Perception - L-Shape-Fitting Augmentations

To determine the impact of the aforementioned augmentations on the quality of the 3D pose estimation, we evaluated the *L-Shape-Fitting* algorithm in several configurations on the categories (*Car*, *Bus*, *Truck*, *Motorcycle*) of the A9 infrastructure dataset. The ablation study results of the *L-Shape-Fitting* augmentation evaluations are presented in Table III.

The ablation study confirms that tracking and historical plausibility alone are not useful to improve over basic *L-Shape-Fitting*. With the addition of the HD map, however, the risk that an earlier bad yaw choice propagates into the future is greatly reduced, and the historical plausibility further increases the gain in mAP from +2.7 to +5.64.

Model	Modality	Fusion Level	Dataset	Precision	Recall	mAP_{3D}
MonoDet3D (Ours)	Image south1	-	A9-I south1	48.12	59.23	49.01
	Image south2	-	A9-I south2	27.84	29.11	26.33
	Image south1+south2	LF	A9-I full	37.98	44.17	37.67
LidarDet3D (Ours)	Point Cloud S+N	EF	A9-I full	8.40	6.32	8.13
	Point Cloud S+N	LF	A9-I full	6.34	5.43	6.10
PointPillars* [27]	Point Cloud N	-	A9-I south1	56.66	57.44	56.10
	Point Cloud N	-	A9-I south2	20.96	31.79	20.62
	Point Cloud S	-	A9-I south2	**36.32**	**48.93**	**35.75**
	Point Cloud S	-	A9-I south1	35.37	50.01	34.81
	Point Cloud S+N	EF	A9-I full	**62.85**	51.22	**62.11**
	Point Cloud S+N	LF	A9-I full	46.97	51.23	46.10
InfraDet3D (Ours):	Image south1 + Point Cloud S+N	LF of (Image + Point Cloud EF)	A9-I south1	**68.83**	**74.89**	(*+12.38*) **68.48**
MonoDet3D + PointPillars	Image south2 + Point Cloud S+N	LF of (Image + Point Cloud EF)	A9-I south2	33.52	44.57	(*-2.54*) 33.21
	Image (south1+south2) + Point Cloud S+N	LF of (Image LF + Point Cloud LF)	A9-I full	38.93	49.94	38.58
	Image (south1+south2) + Point Cloud S+N	LF of (Image LF + Point Cloud EF)	A9-I full	39.28	48.12	38.86

TABLE I: Evaluation results on the A9-I intersection test set (N=North, S=South, EF=Early Fusion, LF=Late Fusion). We report the $mAP_{3D@0.1}$ results for the following six classes: *Car*, *Truck*, *Bus*, *Motorcycle*, *Pedestrian*, *Bicycle*. * PointPillars inference score threshold is set to 0.3.

TABLE II: Ablation study for matched camera-LiDAR detections calculated for south1 camera using early and late fusion. Taking LiDAR detection attributes leads to mAP$_{3D}$ score improvements in all cases.

Fused Attribute	Improvement in mAP_{3D}
Center position	+2.96
Yaw	+0.16
Dimensions	+1.65
Category	+13.10
Total improvement	**+17.87**

TABLE III: Ablation study of *L-Shape-Fitting* (LSF) augmentations on the vehicle category superset of the A9-I dataset.

Configuration	mAP_{3D}	IoU_{3D}
Basic *L-Shape-Fitting* (LSF)	62.01	0.29
LSF with HD map yaw confidence	64.71	0.43
LSF with hist. plausibility via *SORT* tracking	48.42	0.31
LSF with both augmentations	**67.65**	**0.44**

B. Monocular 3D Perception - Performance Considerations

As presented, the monocular 3D object detection pipeline achieves a throughput of 22 FPS in our test bench setup using an RTX 2080S GPU with 1280x720 24-bit RGB input frames. This is limited by the performance of the *YOLOv7* instance segmentation inference time. At 640x480 resolution, the frame rate increases to 66 FPS using *TensorRT*.

C. LiDAR 3D Perception - Runtime Evaluation

Our unsupervised 3D detector achieves a processing speed of 47 FPS as Table IV demonstrates.

Table V shows the runtime of *PointPillars* on the A9-I dataset.

D. Quantitative Results

All four object detection modules were evaluated on the A9-I south1, south2 and full intersection test set (see Table I). The performance on the south1 sub set is 80% higher on average because of better lighting conditions and 2.5x less occlusions. *PointPillars* performs much better on the test set compared to *LidarDet3D* since it was trained on the A9-I

TABLE IV: Runtime evaluation of detector modules on the A9-I test set. All modules are implemented in Python 3.8 and run on a 2.9 GHz dual-core Intel Core i5 CPU.

Module	∅ **Runtime in** ms
Region of Interest Selection	4.05
Ground Segmentation	0.83
Background Filtering	1.05
Outlier Removal	4.82
Clustering	8.00
Bounding Box Fitting	2.15
Classification	0.18
Total runtime	**21.08** (47 FPS)

TABLE V: Runtime evaluation of *PointPillars* on the A9 intersection dataset with a single and registered point clouds. Tested in Python 3.8 and run on a NVIDIA RTX 4090.

Point cloud type	∅ **Runtime in** ms	**FPS**
Single LiDAR point cloud	23.84	42
Registered point cloud	26.11	38

dataset. The south LiDAR has 2.74x more overlapping with the south2 camera which leads to higher mAP value (+15.13), compared to the north LiDAR. Using registered point clouds (early fusion) we achieve the highest results (68.48 mAP) with our *InfraDet3D* fusion model on the A9-I south1 test set by fusing the camera and LiDAR detections on a late fusion level. Table VI demonstrates that among all classes, the *Bus* class exhibits the highest average precision in terms of detection accuracy since it is covered well by all LiDARs.

Class	Car	Truck	Motorcycle	Bus	Pedestrian	Bicycle
Precision	71.75	91.20	82.72	**99.93**	31.37	36.02
Recall	87.33	85.03	70.71	**100.00**	25.49	80.77
AP	71.64	91.03	82.37	**99.93**	30.00	35.93

TABLE VI: Average Precision (AP) results across classes in the A9-I dataset of the best performing *InfraDet3D* model.

E. Qualitative Results

The qualitative results are shown in Figure 6. Note that even objects outside of the sensor's field-of-view, like the black
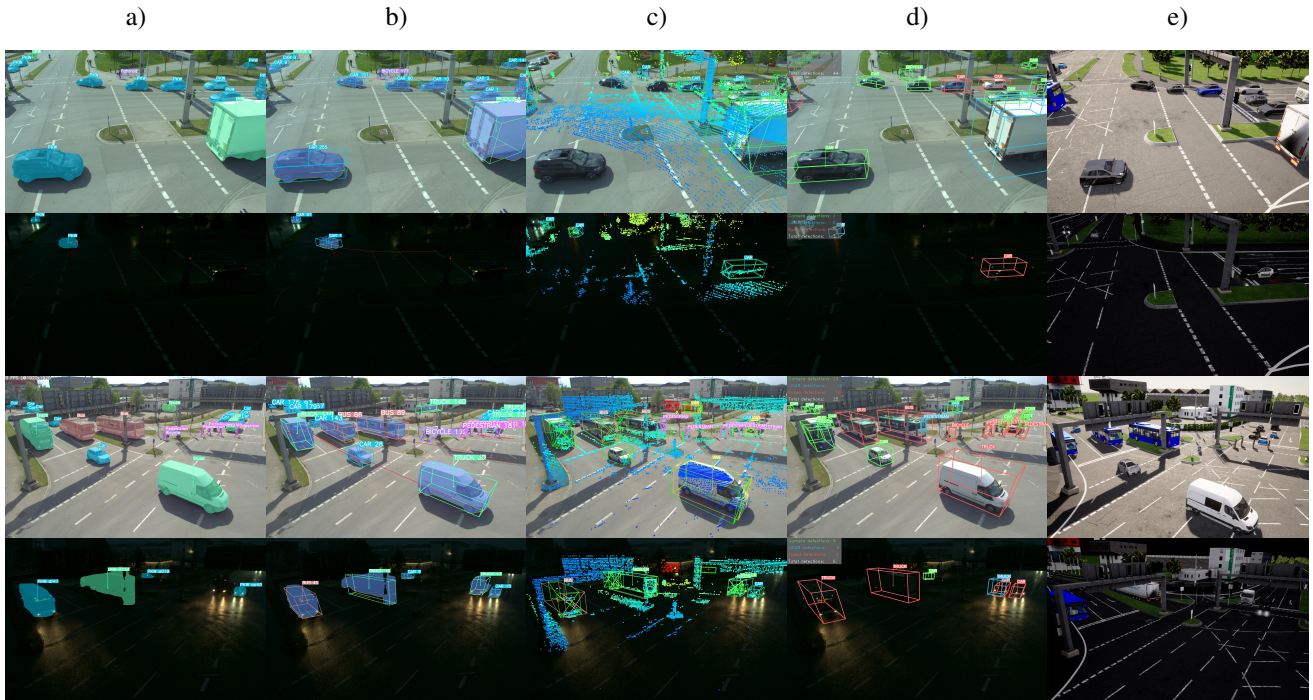
Fig. 6: Qualitative results for south1 camera (first and second row) and south2 camera (third and fourth row). We show the perception results during day time (first and third row) and night time (second and fourth row). The detections are colored by their class color. Column four shows the fusion results in the following colors: green (unmatched camera detections), blue (unmatched LiDAR detections) and red (fused detections). From left to right: a) Instance segmentation, b) *MonoDet3D*, c) *PointPillars*, d) *InfraDet3D*, e) Visualization of the fused perception results in CARLA (using early and late fusion).

car in the first row, can be detected by fusing camera and LiDAR detections. The final perception results are visualized in the CARLA simulation environment, that contains a full reconstruction of the A9 Test Stretch.

IX. Conclusion

InfraDet3D is a novel perception architecture that increases situation awareness and range of traditional single-sensor systems by combining data from multiple sensors distributed on a 20 m long infrastructure gantry bridge. We show that our multi-modal perception framework, fusing multiple roadside LiDARs and cameras, is able to achieve better results ($+1.62$ `mAP`) than object detectors using only the camera input. The distributed sensors combine their perception results and allow to detect partially and even fully occluded objects. Our solution is deployed on high performance edge units and is very cost-effective, since it is distributed among the CPU (calibration, unsupervised point cloud detection, fusion) and the GPU (instance segmentation, supervised detection in point clouds). Future trends and challenges include a better perception in adverse weather conditions such as heavy rain, snow, and fog. These conditions reduce the range, reflection intensity, resolution of point clouds, increase the noise, and produce outliers. In [34] and [35], a method to filter snow points is proposed that will be incorporated in the future. A point cloud compression module will be integrated for real-time communication and data sharing between RSUs and

vehicles. In the future, we plan to extend our framework into a deep fusion architecture. Finally, our goal is to evaluate our models on other infrastructure roadside datasets like DAIR-V2X-I [36], Rope3D [37], LUMPI [38], and IPS300+ [39]. We will also label more roadside sensor data and apply few-shot and active learning [40] to deal with small datasets and limited information. To improve domain adaptation, we will adapt our solution to other roadside LiDAR sensors and different domains (ODDs) to achieve a domain-invariant data representation.

References

[1] Q.-Y. Zhou, J. Park, and V. Koltun, "Fast global registration," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*. Springer, 2016, pp. 766–782.

[2] A. Krämmer, C. Schöller, D. Gulati, V. Lakshminarasimhan, F. Kurz, D. Rosenbaum, C. Lenz, and A. Knoll, "Providentia-a large-scale sensor system for the assistance of autonomous vehicles and its evaluation," *Journal of Field Robotics*, pp. 1156–1176, 2022.

[3] C. Creß and A. Knoll, "Intelligent transportation systems using external infrastructure: A literature survey," *arXiv preprint*, 2021.

[4] W. Zimmer, E. Ercelik, X. Zhou, X. J. D. Ortiz, and A. Knoll, "A survey of robust 3d object detection methods in point clouds," *arXiv preprint arXiv:2204.00106*, 2022.

[5] W. Zimmer, J. Wu, X. Zhou, and A. C. Knoll, "Real-time and robust 3d object detection with roadside lidars," *arXiv preprint arXiv:2207.05200*, 2022.

[6] W. Zimmer, M. Grabler, and A. Knoll, "Real-time and robust 3d object detection within road-side lidars using domain adaptation," *arXiv preprint arXiv:2204.00132*, 2022.

[7] Z. Gong, Z. Wang, B. Zhou, W. Liu, and P. Liu, "Pedestrian detection method based on roadside light detection and ranging," *SAE International Journal of Connected and Automated Vehicles*, vol. 4, no. 12-04-04-0031, pp. 413–422, 2021.

[8] E. Guo, Z. Chen, S. Rahardja, and J. Yang, "3d detection and pose estimation of vehicle in cooperative vehicle infrastructure system," *IEEE Sensors Journal*, vol. 21, no. 19, pp. 21759–21771, 2021.

[9] Z. Bai, G. Wu, X. Qi, Y. Liu, K. Oguchi, and M. J. Barth, "Cyber mobility mirror for enabling cooperative driving automation in mixed traffic: A co-simulation platform," *IEEE Intelligent Transportation Systems Magazine*, pp. 2–15, 2022.

[10] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in *2017 IEEE international conference on image processing (ICIP)*. IEEE, 2017, pp. 3645–3649.

[11] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11621–11631.

[12] C. Creß, W. Zimmer, L. Strand, M. Fortkord, S. Dai, V. Lakshminarasimhan, and A. Knoll, "A9-dataset: Multi-sensor infrastructure-based dataset for mobility research," in *2022 IEEE Intelligent Vehicles Symposium (IV)*, 2022, pp. 965–970.

[13] W. Zimmer, A. Rangesh, and M. Trivedi, "3d bat: A semi-automatic, web-based 3d annotation toolbox for full-surround, multi-modal data streams," in *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2019, pp. 1816–1821.

[14] Y. Chongjian, X. Liu, X. Hong, and F. Zhang, "Pixel-level extrinsic self calibration of high resolution lidar and camera in targetless environments," *IEEE Robotics and Automation Letters*, vol. PP, pp. 1–1, 07 2021.

[15] R. Ranftl, K. Lasinger, D. Hafner, K. Schindler, and V. Koltun, "Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 3, pp. 1623–1637, 2022. [Online]. Available: https://doi.org/10.1109/TPAMI.2020.3019967

[16] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679–698, 1986.

[17] Daniel, "Point cloud upsampling," visited on 13/1/2022. [Online]. Available: https://github.com/danielTobon43/upsamplingCloudPCL

[18] X. Liu, C. Yuan, and F. Zhang, "Targetless extrinsic calibration of multiple small fov lidars and cameras using adaptive voxelization," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–12, 2022.

[19] X. Zhang, W. Xu, C. Dong, and J. M. Dolan, "Efficient l-shape fitting for vehicle detection using laser scanners," in *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2017, pp. 54–59.

[20] L. Gressenbuch, K. Esterle, T. Kessler, and M. Althoff, "Mona: The munich motion dataset of natural driving," in *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2022, pp. 2093–2100.

[21] M. Rezaei, M. Azarmi, and F. M. P. Mir, "Traffic-net: 3d traffic monitoring using a single camera," *arXiv preprint arXiv:2109.09165*, 2021.

[22] J. Carrillo and S. Waslander, "Urbannet: Leveraging urban maps for long range 3d object detection," in *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2021, pp. 3799–3806.

[23] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," *arXiv preprint arXiv:2207.02696*, 2022.

[24] E. Schubert, J. Sander, M. Ester, H. P. Kriegel, and X. Xu, "Dbscan revisited, revisited: why and how you should (still) use dbscan," *ACM Transactions on Database Systems (TODS)*, vol. 42, no. 3, pp. 1–21, 2017.

[25] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking," in *2016 IEEE international conference on image processing (ICIP)*. IEEE, 2016, pp. 3464–3468.

[26] T. Zhang and P. J. Jin, "Roadside LiDAR vehicle detection and tracking using range and intensity background subtraction," *Journal of Advanced Transportation*, vol. 2022, pp. 1–14, apr 2022. [Online]. Available: https://doi.org/10.1155%2F2022%2F2771085

[27] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 12697–12705.

[28] O. D. Team, "Openpcdet: An open-source toolbox for 3d object detection from point clouds," https://github.com/open-mmlab/OpenPCDet, 2020.

[29] R. Jonker and T. Volgenant, "A shortest augmenting path algorithm for dense and sparse linear assignment problems," in *DGOR/NSOR: Papers of the 16th Annual Meeting of DGOR in Cooperation with NSOR/Vorträge der 16. Jahrestagung der DGOR zusammen mit der NSOR*. Springer, 1988, pp. 622–622.

[30] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[31] D. F. Crouse, "On implementing 2d rectangular assignment algorithms," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 52, no. 4, pp. 1679–1696, 2016.

[32] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *2009 IEEE international conference on robotics and automation*. IEEE, 2009, pp. 3212–3217.

[33] P. J. Besl and N. D. McKay, "Method for registration of 3-d shapes," in *Sensor fusion IV: control paradigms and data structures*, vol. 1611. Spie, 1992, pp. 586–606.

[34] M.-H. Le, C.-H. Cheng, D.-G. Liu, and T.-T. Nguyen, "An adaptive group of density outlier removal filter: Snow particle removal from lidar data," *Electronics*, vol. 11, no. 19, p. 2993, 2022.

[35] J.-I. Park, J. Park, and K.-S. Kim, "Fast and accurate desnowing algorithm for lidar point clouds," *IEEE Access*, vol. 8, pp. 160202–160212, 2020.

[36] H. Yu, Y. Luo, M. Shu, Y. Huo, Z. Yang, Y. Shi, Z. Guo, H. Li, X. Hu, J. Yuan *et al.*, "Dair-v2x: A large-scale dataset for vehicle-infrastructure cooperative 3d object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 21361–21370.

[37] X. Ye, M. Shu, H. Li, Y. Shi, Y. Li, G. Wang, X. Tan, and E. Ding, "Rope3d: The roadside perception dataset for autonomous driving and monocular 3d object detection task," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 21341–21350.

[38] S. Busch, C. Koetsier, J. Axmann, and C. Brenner, "Lumpi: The leibniz university multi-perspective intersection dataset," in *2022 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2022, pp. 1127–1134.

[39] H. Wang, X. Zhang, Z. Li, J. Li, K. Wang, Z. Lei, and R. Haibing, "Ips300+: a challenging multi-modal data sets for intersection perception system," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 2539–2545.

[40] A. Hekimoglu, M. Schmidt, A. Marcos-Ramiro, and G. Rigoll, "Efficient active learning strategies for monocular 3d object detection," in *2022 IEEE Intelligent Vehicles Symposium (IV)*, 2022, pp. 295–302.