TUM School of Computation, Information and Technology
Technical University of Munich

Vehicle-Infrastructure Cooperative 3D Object Detection to Support Autonomous Driving Functions

Kooperative 3D-Objekterkennung zwischen Fahrzeug und Infrastruktur zur Unterstützung von autonomen Fahrfunktionen

Supervisor Prof. Dr.-Ing. habil. Alois C. Knoll

Advisor Walter Zimmer, M.Sc.

Author Suren Sritharan

Date January 15, 2024 in Munich

Disclaimer

I confirm that this interdisciplinary project is my own work and I have documented all sources and material used.

Munich, January 15, 2024

(Suren Sritharan)

Abstract

In the context of autonomous driving applications, enhancing the robustness and accuracy of 3D vehicle detection has become increasingly important. Traditional single-viewpoint sensors face challenges like limited field of view and occlusion, hindering accurate object detection. Cooperative perception aims to overcome the constraints of standalone sensors by aggregating information from multiple perspectives, reducing blind spots, and enhancing reliability in complex scenarios. This project introduces CMTCoop, a transformer-based deep fusion model for cooperative 3D object detection. The proposed model adapts the architecture of the Cross-Modal Transformer for cooperative 3D object detection and employs a deep fusion methodology to enable perception from both infrastructure and vehicular perspectives. Through experiments, this work demonstrates that cooperative perception improves 3D detection mAP by +9.96% compared to vehicle-only perception. Further evaluation shows that the model suffers a drop in speedup (from 8 FPS to 4.5 FPS) due to increased complexity, and future research directions to improve efficiency are also presented.

Zusammenfassung

Die Verbesserung der Robustheit und Genauigkeit der 3D-Fahrzeugerkennung wird im Zusammenhang mit autonomen Fahranwendungen immer wichtiger. Herkömmliche Ein-Punkt-Sensoren stehen vor Herausforderungen wie einem begrenzten Sichtfeld und Verdeckungen, die eine genaue Objekterkennung behindern. Kooperative Wahrnehmung zielt darauf ab, die Einschränkungen von Einzelsensoren zu überwinden, indem sie Informationen aus mehreren Perspektiven zusammenführt, tote Winkel reduziert und die Zuverlässigkeit in komplexen Szenarien erhöht. In diesem Projekt wird CMTCoop vorgestellt, ein transformatorbasiertes Deep-Fusion-Modell für die kooperative 3D-Objekterkennung. Das vorgeschlagene Modell adaptiert die Architektur des Cross-Modal Transformers für die kooperative 3D-Objekterkennung und verwendet eine Deep-Fusion-Methode, um die Wahrnehmung sowohl aus der Infrastruktur- als auch aus der Fahrzeugperspektive zu ermöglichen. Durch Experimente zeigt diese Arbeit, dass die kooperative Wahrnehmung die 3D-Erkennung mAP um +9,96% im Vergleich zur reinen Fahrzeugwahrnehmung verbessert. Eine weitere Bewertung zeigt, dass das Modell aufgrund der erhöhten Komplexität einen Geschwindigkeitsverlust erleidet (von 8 FPS auf 4,5 FPS), und zukünftige Forschungsrichtungen zur Verbesserung der Effizienz werden ebenfalls vorgestellt.

Contents

Chapter 1

Introduction

Autonomous driving applications have seen a significant rise in interest recently. With the increased use of automated vehicles, enhancing the robustness and accuracy of 3D vehicle detection has become vital to ensure safety under varying conditions. As vehicles become more sophisticated in their sensory capabilities, leveraging the collective intelligence of multiple perception units has emerged as a promising approach to address the challenges associated with occlusions, varying environmental conditions, and real-time decision-making.

1.1 Motivation

Traditionally, autonomous vehicles equipped with different onboard sensors have been used for decision-making processes involved in navigation. The type of sensors include camera / LiDARs, radars, and GPS/IMU sensors, often covering a 360^o view around the vehicle. However, these sensors are often positioned on the top/side of the vehicles, which limits their field of view, and their perception is often hindered due to objects around them, especially larger vehicles, which can occlude their surroundings, as shown in Figure 1.1. This leads to many challenges in detecting 3D objects around the ego-vehicle, which is detrimental to vital downstream tasks such as navigation and control.

In addition to this, infrastructure perspective-based vehicular object detection is also used for traffic analysis, accident detection, and the creation of digital twins. While this approach overcomes issues caused by occlusion, the sensor data's accuracy is lower than the sensor data obtained from the ego vehicle, and it decreases as the distance between the infrastructure

Figure 1.1: Vehicular perspective (left) and infrastructure perspective (right) camera images. The ego vehicle (marked in red) is covered by larger vehicles (marked in green), which obscures its perception in detecting other objects (marked in blue).

sensor and the ego vehicle increases. Furthermore, for vehicular navigation, the ego-vehicle has to be in the range of the infrastructure sensor for the detections to be transmitted. As such, using a combination of these sensors for object detection would be beneficial.

1.2 Cooperative perception

Cooperative perception in the context of 3D object detection involves coordinating sensor data from diverse sources, such as LiDAR, radar, and cameras, across a network of connected vehicles or infrastructure. This paradigm not only strives to improve the accuracy of individual vehicle perception systems but also aims to provide a holistic understanding of the surrounding environment by aggregating information from multiple perspectives.

The key motivation behind cooperative perception lies in overcoming the limitations of traditional standalone sensor configurations. By pooling information from neighboring vehicles or infrastructure, collaborative perception systems can mitigate blind spots, reduce false positives, and enhance the overall reliability of 3D vehicle detection. This collaborative approach is particularly valuable in complex traffic scenarios, urban environments, and situations where a single sensor's viewpoint may be insufficient to make informed decisions.

1.3 Proposed model

This project proposes CMTCoop, a transformer-based deep fusion model for cooperative perception. The proposed model is based on the Cross-Modal Transformer proposed in [Yan+23a] and modifies the architecture to enable cooperative perception from infrastructure and vehicular perspectives. In addition, this work also details the preprocessing steps used to prepare the cooperative perception data for training and evaluation.

Through extensive experiments, this work shows that cooperative perception improves object detection performance compared to vehicle-only perception models. Furthermore, the proposed model improves the efficacy of object detection compared to the current SOTA model proposed in [24]. In addition, the efficiency and model complexity are compared with other existing works.

1.4 Limitations

V2X communication has been studied in detail recently and poses many challenges [Hua+23] concerning data transmission, sensor synchronization, and processing power limitations. In terms of data transmission, one must consider the communication delay that occurs during transmission and the bandwidth and the size of data that needs to be transferred. These two factors correlate where larger data transmission leads to longer total transmission time. Different studies are being conducted to reduce the message size and develop different V2X protocols to enhance transmission efficiency.

Sensor synchronization is another issue in collaborative perception. This has been studied for vehicular-only or infrastructure-only sensor data collection. However, in the case of collaborative perception, vehicular and infrastructure sensors must be synchronized together to have the same timestamps, and the lack of a shared clock has also led to studies in alternative methodologies.

Compared to the aforementioned works, this study only focuses on offline 3D object detection. As such, other issues related to synchronization and communication delays have been disregarded for brevity, and this work assumes that the data is obtained without any delay or loss.

Chapter 2

Related work

There are a myriad of tasks in the domain of autonomous driving, including 2D, object detection, 3D object detection, object tracking, shape reconstruction, and trajectory prediction. However, this project focuses on 3D object detection, and as such, this chapter will focus on the recent and state-of-the-art models for this task.

Multiple sensors are used to collect the data for object detection, with cameras and LiDARs often being the majority, and these sensors are mounted either on roadside infrastructures or on vehicles. Different models use different combinations of these data for object detection. As such, this chapter first discusses the related works separated by the type of sensors used and then the viewpoints used by different models.

2.1 Sensor-based model classification

Cameras and LiDARs are the most common types of sensors used for 3D object detection. The two sensors complement each other, wherein cameras have the advantage of being low-cost devices and provide color and texture information. On the other hand, LiDARs are robust to different weather conditions and provide depth information that is absent in RGB images. As such, certain models use only images taken from cameras, whereas others use LiDAR point clouds for object detection. Finally, the data from the two sensors can be fused together to perform fusion-based object detection, and the prior works in each of these approaches are discussed below.

2.1.1 Camera-based models

Camera-only models were the pioneers in object detection due to their low cost and high availability of datasets. Early approaches to monocular 3D object detection closely resembled their 2D counterparts [BL19; Sim+19], where the 3D bounding boxes are predicted based on the 2D localization information in a two-step process. Later, certain works took inspiration from the advanced 2D detectors to directly predict 3D bounding boxes for individual objects [Wan+21]. However, in the presence of multiple cameras, the predictions are fused from different perspectives, and redundant detections are eliminated. This process is often complex, and as a result, such late-fusion approaches often produce inferior results.

Another approach involves transforming image features from an image perspective into Bird's Eye View (BEV) features, predicting 3D bounding boxes from a top-down perspective. Different methods achieve this by transforming image features into BEV features with depth information. The BEV features can then be fused together from multiple camera views. LSS

[PF20] proposed a model whereby each image feature is lifted individually into a frustum of features for each camera and then "splat" all frustums into a rasterized bird's-eye-view. BEVDet [Hua+21] improves this performance through an exclusive data augmentation strategy and improved post-processing. BEVDepth [Li+23] further improves the depth prediction compared to LSS, leading to better prediction accuracy.

The introduction of transformers [Car+20] has also inspired many works in this realm. DETR3D [Wan+22b] and BEVFormer [Li+22b] project the predefined BEV queries onto images and then employ the transformer attention to model the relation of multi-view features. The above methods explicitly project the local image feature from 2D perspective view to BEV. In contrast, PETR [Liu+22] and SpatialDETR [Dol+22] take a different approach by incorporating positional embeddings dependent on camera poses. This allows transformers to implicitly learn the projection from image views to 3D space.

2.1.2 LiDAR-based models

LiDAR-based models became popular later, offering robustness, particularly in challenging weather conditions and low-light scenarios. LiDAR-based 3D object detection models aim to predict the 3D bounding boxes by leveraging point clouds obtained from LiDAR sensors. Existing methods adopt various strategies to process the point cloud into different representations. Point-based approaches directly derive features from raw point clouds and subsequently predict 3D bounding boxes. Notably, PointNet [Qi+17] introduced the pioneering concept of processing the point cloud end-to-end.

In contrast, alternative methods involve projecting the LiDAR point clouds onto different feature spaces for further processing. 3D voxels-based approaches were the pioneers in this regime, dividing the dense point clouds into sparse voxels [ZT18; YML18] for feature extraction. For example, VoxelNet [ZT18] initially divides the raw point clouds into regular voxel grids and then employs the PointNet network to extract features from the points within each voxel grid. Point clouds are also projected to pillar features [Lan+19; Wan+20], which use lower dimensional representational features of point clouds to enable faster point processing. PointPillars [Lan+19], compresses point clouds into BEV space, representing each “pillar” with a feature. In [Fan+21; Sun+21] point clouds are projected to range images, followed by 2D convolution on the images for object detection.

2.1.3 Camera-LiDAR fusion models

Fusion models employ the data obtained from both cameras and LiDARs for object detection tasks. Single-viewpoint fusion models employ either vehicular cameras and LiDAR [Liu+23; Vor+20; Yan+23a] or infrastructure cameras and LiDAR [Zim+23a] for 3D object detection. These models, which combine information from both images and point clouds, have demonstrated superior performance compared to previous methods [Zim+23a].

2.2 Viewpoint based classification

The models discussed above can again be classified based on the viewpoint for training. Though these works can be extended to support the alternative viewpoint models (i.e., vehicular perspective models can be retrained on infrastructure data to perform infrastructure viewpoint-based object detection), this section briefly describes each model type.

2.2.1 Vehicular viewpoint models

These models use only the data collected from vehicular onboard sensors. Due to the wide availability of such datasets [Gei+13; Cae+20; Sun+20], vehicular viewpoint models are more common. Camera-only vehicular viewpoint models have been proposed in [Kum+22; Wu+23] and LiDAR-based models have been proposed in [Lan+19].

2.2.2 Infrastructure viewpoint models

Infrastructure viewpoint models use the data collected from roadside sensor units (RSU) for object detection. Due to the complexity of building the infrastructure for data collection, such models are sparse compared to the vehicular viewpoint models. Both infrastructure camera-based models [Yan+23b] and LiDAR-based models [Zim+23d] exist in literature.

2.2.3 Cooperative perception models

Using data from multiple viewpoints, cooperative perception models have proven effective in mitigating occlusion-related challenges prevalent in vehicular sensor-based models. V2I cooperative perception models [Bai+23; He+21; Bai+22; Xu+22b; Yu+23a; Wei+23] integrate sensor data from both vehicles and infrastructure, while V2V models [Hu+23; Xu+22a; QZ23] facilitate communication of sensor data between two ego-vehicles.

Previous works have shown that deep cooperative sensor fusion models outperform all other configurations of data (camera vs. LiDAR vs. multi-modal), viewpoint (single vs. multi-viewpoint), and fusion (early vs. late vs. deep) methods. V2X-ViT[Xu+22b] and CoBEVT [Xu+22a] use transformers for cooperative perception with intermediate feature fusion. CoBEVFusion [QZ23] further extends these works with Dual Window-based Cross-Attention for deep feature fusion in the BEV space to improve performance. QUEST [Fan+23] achieves cooperative perception by streaming the transformer queries from the infrastructure side to the vehicle side when triggered by the vehicular model. To this end, this work also proposes a multi-modal cooperative deep fusion model for 3D object detection.

Chapter 3

Dataset and Tools

This chapter discusses the datasets available for object detection in autonomous driving scenarios, including the specific dataset used for this project. It also describes the tools that were used in various stages of the project.

3.1 Dataset

With the rise of autonomous driving research, a wide variety of datasets have been introduced for various tasks, including 3D vehicular object detection, reconstruction, tracking, forecasting, etc. This work focuses mainly on 3D object detection, and the datasets for this are mainly categorized based on the viewpoint. Table 3.1 summarizes some of the main features of a few selected datasets, and each of these is described in detail in the following subsections.

3.1.1 Single viewpoint dataset

Single viewpoint datasets are obtained from a single point of reference, either an ego-vehicle or roadside infrastructure. Onboard sensor-based datasets encompass diverse sensor data collected from a moving vehicle equipped with high-resolution cameras, LiDARs, radars, and GPS/INS systems. These abundant datasets provide extensive annotated data, including bounding boxes, track IDs, segmentation masks, and depth maps, all collected across various

Table 3.1: Comparison of different 3D autonomous driving datasets. Best metric values in each category are highlighted in bold, and second-best are underlined.

Name	Year	View	# Point Clouds	# Images	# Classes	# 3D Boxes	Track IDs
KITTI [Gei+13]	2013	onboard	15 k	15 k	8	80 k	-
nuScenes[Cae+20]	2020	onboard	400 k	1,400 k	23	1,400 k	✓
Waymo [Sun+20]	2020	onboard	200 k	1,000 k	4	12,600 k	✓
IPS300+ [Wan+22a]	2022	roadside	14 k	14 k	7	4,541 k	-
Rope3D [Ye+22]	2022	roadside	-	50 k	13	1,500 k	-
DAIR-V2X-I [Yu+22]	2022	roadside	10 k	10 k	10	493 k	-
TUMTraf Intersection [Zim+23c]	2023	roadside	4.8 k	4.8 k	10	62 k	✓
V2XSet [Xu+22b]	2022	V2V&I	11 k	44 k	1	233 k	-
V2X-Sim [Li+22a]	2022	V2V&I	10 k	60 k	1	26 k	✓
V2V4Real [Xu+23]	2023	V2V	20 k	40 k	5	240 k	✓
DAIR-V2X-C [Yu+22]	2022	V2I	39 k	39 k	10	464 k	-
DAIR V2X-Seq (SPD) [Yu+23b]	2023	V2I	15 k	15 k	9	10.5 k	✓
TraffiX Coop. V2X Dataset	2023	V2I	1.6 k	4 k	8	50 k	✓

urban driving scenarios. A few well-known examples of such datasets are KITTI [Gei+13], nuScenes [Cae+20], and Waymo [Sun+20].

On the other hand, infrastructure viewpoint datasets use the data obtained from roadside sensor units (RSUs), and they are less abundant than vehicular viewpoint datasets due to the complexities of building RSUs. Noteworthy multi-modal datasets incorporating camera and LiDAR data are presented in [Cre+22; Zim+23c; Bus+22]. These datasets, obtained from Infrastructure Perception Systems (IPS), offer high-quality information. Additionally, [Ye+22] provides a dataset consisting solely of images captured from different viewpoints and under diverse traffic conditions. These datasets offer a top-down view of crowded intersections under varying conditions, overcoming issues like occlusions caused by other vehicles, which is a prevailing issue in vehicular viewpoint datasets. Consequently, they tend to have a higher number of object labels compared to onboard sensor-based datasets.

Datasets obtained from a vehicular viewpoint are commonly preferred since vehicles only sometimes have the opportunity to get additional information from surrounding RSUs or other vehicles. As such, the ego vehicle must act independently when making decisions. On the other hand, infrastructure viewpoint data provides information from a different viewpoint, and the higher elevation of such sensors provides a larger field of view. However, the distance from the ego vehicle often results in lower data accuracy, especially when considering nearby objects. Thus, a combination of both these data is beneficial for decision-making.

3.1.2 Cooperative perception dataset

Cooperative perception datasets leverage information from multiple viewpoints to gain a more comprehensive understanding of environments. This helps overcome limitations associated with single viewpoint datasets that were discussed, such as occlusion, limited field of view (FOV), low point cloud density, and limited accuracy. Cooperative perception datasets are also called V2X datasets since the information must be communicated to the vehicle from different sources. In this sense, the datasets can be divided into V2V datasets, where the data is transmitted between multiple ego vehicles, and V2I datasets, where communication occurs between an ego vehicle and an RSU.

The DAIR-V2X dataset family [Yu+22] stands out as a prominent and largest cooperative multi-modal **V2I dataset** available. It consists of three subsets: an intersection dataset, a vehicle dataset, and a cooperative dataset. The V2X-Seq dataset [Yu+23b] extends selected sequences from the DAIR-V2X dataset with track IDs. It is divided into a sequential perception dataset (SPD) designed for tracking tasks and a forecasting dataset focusing on trajectory forecasting. V2V4Real [Xu+23] proposes a multi-modal cooperative dataset focusing exclusively on **V2V perception**. This dataset, similar in size to other cooperative datasets, features two vehicles equipped with cameras, LiDAR, and GPS/IMU integration systems, collecting multi-modal sensor data for diverse scenarios. These datasets, however, have limitations, such as their restricted global availability and fewer labeled objects. As such, an alternative dataset proposed in [24] is used for this project.

Collecting a large amount of data for a cooperative perception dataset is challenging compared to single viewpoint datasets. Since the sensor nodes (vehicle or infrastructure) should be within communicable and visual distance of each other, cooperative datasets are smaller in size with fewer annotations. To overcome this challenge, Simulated multi-agent perception datasets are introduced in [Li+22a; Xu+22c; Xu+22b]. Utilizing simulators such as CARLA [Dos+17], SUMO [Lop+18], and OpenCDA [Xu+22c], these datasets incorporate multi-modal sensor data (camera and LiDAR) obtained from multi-agent sensor recordings from roadside units (RSUs) and multiple ego vehicles, enabling collaborative perception. In addition to images and point clouds, these datasets include additional data such as depth

maps, HD maps, pixel-wise and point-wise semantic labels, etc. However, their utility is currently limited due to the simulated nature of the data, and their extendability to real-life applications is limited.

3.2 TUMTraf Dataset Family

The TUMTraf dataset overcomes many of the limitations discussed previously, and as such, it is used in this project. The TUMTraf dataset family consists of two datasets: the TUMTraf intersection dataset [Zim+23c] and the TUMTraf cooperative dataset [24]. The TUMTraf intersection dataset contains infrastructure-only camera and LiDAR data for 3D object detection. In contrast, the TUMTraf cooperative dataset is a V2I cooperative dataset containing both vehicular and infrastructure, image, and point cloud data with approximately 25k labeled bounding boxes and GPS/IMU data.

Since this project focuses on cooperative object detection, this section focuses mainly on the TUMTraf cooperative dataset. However, the TUMTraf intersection dataset is used to benchmark the proposed model's infrastructure-only performance. Furthermore, both these models are used for transfer learning, wherein the cooperative perception model is pretrained on the TUMTraf intersection dataset and then fine-tuned on the TUMTraf cooperative dataset.

3.2.1 Data collection

The test bed for data collection is located at the S110 intersection of the A9 test field belonging to the Providentia++ project. The east side of the intersection has a gantry bridge on which various sensors are mounted for data collection. This includes multiple cameras, LiDARs, radars, and event-based cameras located above the street level. However, for this project, only the LiDARs and cameras were used. In addition to these infrastructure sensors, onboard vehicular sensors are mounted on the roof of an Audi A1 vehicle. Both these sets of sensors together were used to generate the TUMTraf cooperative dataset.

The Basler ace acA1920-50gc cameras with a resolution of 1920x1200 were used to collect the RGB images from the intersection and vehicle. The intersection contains three cameras, namely south 1, south 2, and north, with an 8mm lens, and a single camera is mounted on the vehicle with a 16 mm lens. One Ouster OS1-64 LiDARs was used on the infrastructure side, with a range of 120 m and a field of view of $360^o \times 45^o$. A Robosense RS-LiDAR-32 is mounted on the vehicle, with a range of 150 m and a field of view of $360^o \times 70^o$.

The data was collected through multiple runs of 10 s each on the aforementioned test bed. Then, five scenes consisting of 100 frames each were selected from the recorded data based on the distribution of vehicles and the variety of scenes.

3.2.2 Point cloud registration

After collecting the dataset, the first step is to combine the vehicular point clouds and image point clouds to begin labeling. The process involves finding the projection matrix between the two viewpoints. Theoretically, this can be done if the relative position and orientation of the infrastructure and vehicular LiDARs are known. However, in practice, the measurements may contain errors that can be accumulated over the process, and thereby the registered point clouds could be erroneous. As such, point cloud registration is done in two stages. First, a set of keyframes were chosen from all the frames, and the point cloud fusion was

automated using point-to-point ICP. Then, manual adjustments were made by hand to fix the errors in these keyframes, and frames in between were registered through interpolation.

3.2.3 Data preparation and annotation

Once the point clouds were registered, noisy points were removed based on their intensities and 3D location. Next, images were undistorted based on the intrinsic camera distortion parameters. Finally, the set of 500 frames was divided among five expert annotators to manually label the 3D locations, dimensions, and classes of the objects. The labeling process involves referencing the registered vehicle-infrastructure point cloud to draw the 3D bounding boxes and is guided by the corresponding multiview images. This process generates annotation files in the final ASAM OpenLABEL format label files. The annotations were then rechecked, and the following statistics were then generated.

3.2.4 Data statistics

The data distribution of the TUMTraf Intersection dataset and TUMTraf cooperative dataset are briefly described in this section. Further details can be found in the original paper.

TUMTraf intersection dataset

The TUMTraf Intersection Dataset is comprised of 2,400 data frames. Each frame has two LiDAR point clouds, two camera images, and an associated label file. The dataset is divided into train, validation, and test sets following an 80-10-10 split. Consequently, the train set comprises 1,920 data frames, the validation set includes 240 data frames, and the test set consists of 240 data frames. The data split is done using stratified sampling, which divides the dataset aiming to support robust model training and evaluation across distinct subsets of the dataset.

The data distribution of the train, test, and validation sets from stratified sampling are shown in Figure 3.1. The InfraDet3D model was later used as a benchmark for this dataset.

TUMTraf cooperative dataset

The initial version of the TUMTraf Cooperative Dataset consisting of 500 data frames was used for this project. The dataset was later extended with 300 more frames, including nighttime data, and these additional frames are used to study the extendability of the proposed model to unseen sequences. Each frame incorporates two LiDAR point clouds (infrastructure and vehicle), three infrastructure camera images, one vehicle camera image, and an accompanying label file containing the object annotation and vehicle-to-infrastructure point cloud transformation matrix.

Similar to the TUMTraf Intersection dataset, the entire dataset was split into training, validation, and test sets. An 80-10-10 split was used for this purpose, and consequently, the training, validation, and test sets contain 400, 50, and 50 samples, respectively. In this step, a specific challenge is that the dataset is biased, and random sampling would lead to biased stratification. As such, a multi-label stratified sampling technique is used to ensure that the train, validation, and test sets have a similar class distribution. The data distribution of each set is shown in Figure 3.2.

A notable difference between this dataset and the TUMTraf intersection dataset is that, after pre-processing the OTHER, and EMERGENCY_VEHICLE classes have been ignored due

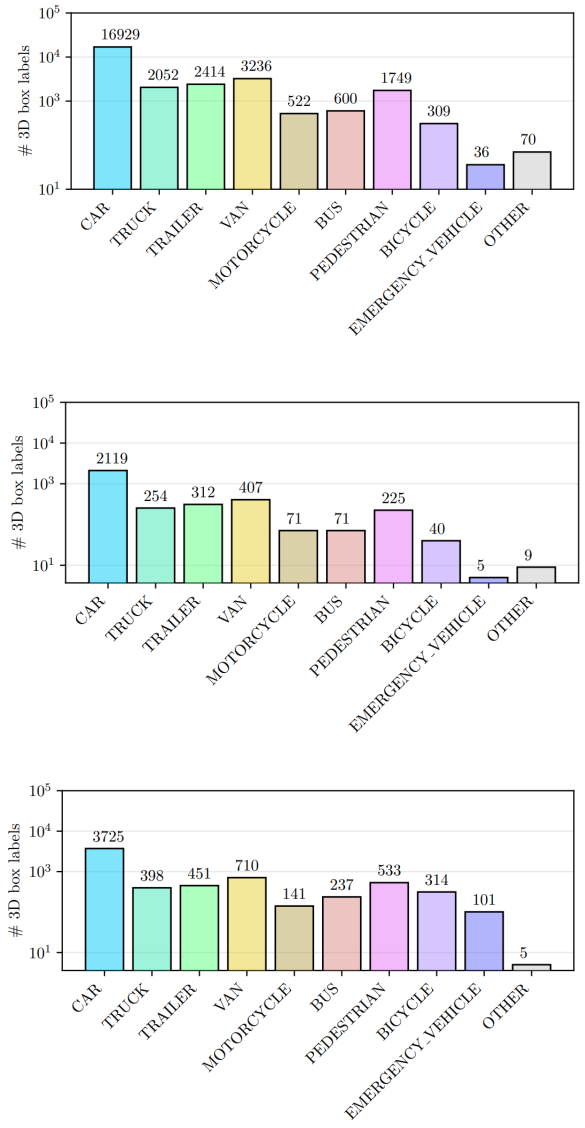

Figure 3.1: Class distribution of the train, validation, and test sets of the TUMTraf Intersection dataset.

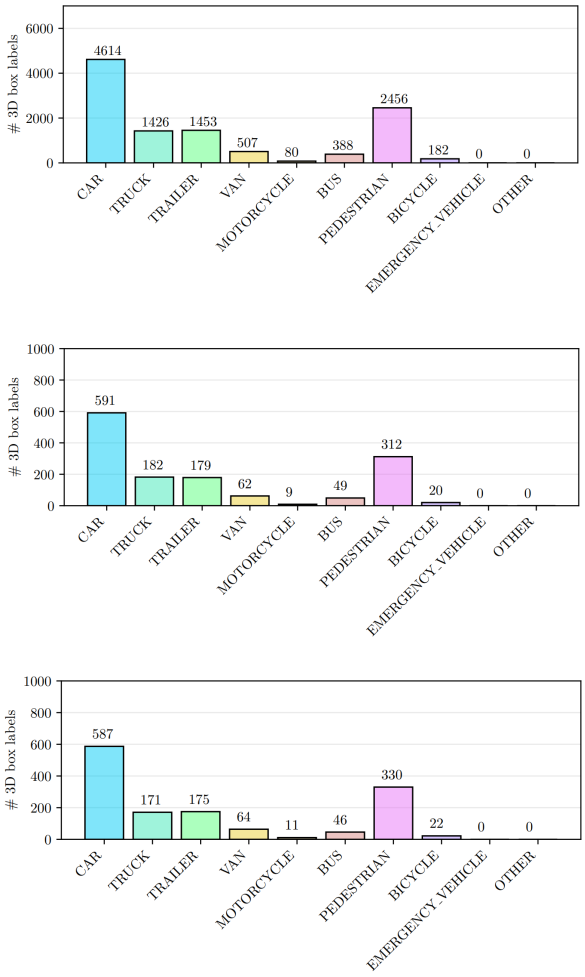

Figure 3.2: Class distribution of the train, validation, and test sets of the TUMTraf cooperative dataset.

to their absence. Furthermore, the MOTORCYCLE class is also ignored since the number of instances is very low compared to other classes.

3.3 Tools

Various tools were used in different stages of the data collection, preparation, labeling, and model development and training stages. These tools are listed in this section.

3.3.1 Data collection and preparation tools

The ROS driver was used in the sensors used for collection and the sensors were synchronized to a central NTP server. Point cloud registration was automated using Open3D, and manual registration and visualization were done through Blender. In the case of data preparation, the TUMTraf devkit [Tea23] was used for point cloud noise removal and image undistortion.

3.3.2 Data annotation tool

The proAnno [24] annotation tool which is based on 3D BAT [ZRT19] was used for data annotation. It supports multi-modal data visualization of point cloud and image data from vehicular and infrastructure perspectives. This allows for a much wider field of view of providing detailed information when labeling the objects. Furthermore, proanno also allows adding other information, such as object ID, which allows temporal tracking and occlusion levels of different objects. However, this information was not required for the task of 3D object detection and, as such, was not added during the labeling process.

3.3.3 Model training and evaluation tools

The model is implemented based on the MMDetection3D frameworks [Con20], and the data augmentation and model training are carried out based on this framework. The augmentations used are explained in Section 4.2 and the model architecture in Section 4.3.2. The MMDetection3D framework also provides an evaluation script to measure the efficacy in terms of the BEV mAP. In addition to this, the frames per second (FPS) is used as a measure of efficiency, which was implemented as an extension of this evaluation script. Qualitative evaluation was performed using the TUMTraf devkit [Tea23], which also provides the implementation for 3D mAP as an additional measure of efficacy. Finally, the `nvidia-smi` command was used to measure the memory consumption of the model. The evaluation metrics specified here are detailed in Section 5.1.

Chapter 4

Methodology

This chapter explains the model architecture, data preparation, and preprocessing steps in detail.

4.1 Data pre-processing

The data preprocessing involves two stages: (1) Data split and (2) Label conversion. The data split was shown in Section 3.2.4, and the label conversion method is explained here. In addition to this, the TUMTraf dataset also needs a custom data loader, as specified in the `mmdetection3D` standard, for it to be used for training and evaluation purposes. These steps are explained in detail below.

4.1.1 Data conversion

The nuScenes data format used by CMT and many other models differs from the file formats used by the TUMTraf Cooperative dataset. While the images have the same datatype (.png) in these two datasets, the format of the point clouds and labels differ. As such, the point clouds are first converted from the .pcd file format to .bin format to make them compatible with the nuScenes format, which is used to train the CMT model.

Next, the information about the train, test, and validation sets must be saved in a pickle (.pkl) file with the format shown in Listing 4.1. The pickle file is a list where each element in the list corresponds to the files corresponding to one timestamp. In addition to the timestamp, each element has the keys for the LiDAR files, namely the 'vehicle_lidar_path', the 'infrastructure_lidar_path', and the 'registered_lidar_path', which contains the location of each of the bin files created in the previous steps. In addition, the sweeps used by the LiDAR could be included, but this isn't relevant for the LiDARs used for the TUMTraf dataset family. Next, the 'lidar_anno_path' represents the path to the JSON file containing the annotation information. The 'vehicle2infrastructure' key contains the 3x4 transformation matrix from the vehicular LiDAR to the infrastructure LiDAR. Next, the vehicular and infrastructure camera information is included in the 'vehicle_cams' and 'infrastructure_cams.' As specified before, the vehicle has a single camera, and the infrastructure has three cameras. Each camera has the 'data_path' key, which specifies the location of the image. In addition, the 'camera_intrinsics' and the 'sensor2lidar' contain a 3x3 camera intrinsic matrix and a 3x4 transformation matrix from the camera to the corresponding LiDAR sensor for each camera. Finally, the information about the labels is stored in the remaining keys. The 'gt_boxes' contain an $N_t \times 7$ matrix specifying the position, size and orientation of each bounding box. 'gt_names' is an array of size

N_t containing the class names. 'num_lidar_pts' specifies the number of lidar points in each of these bounding boxes. 'num_radar_pts' is always set to zero since radars are not used in this data, and 'valid_flag' is always set to True since all labels are used.

```
{
'infos': [
{
    'timestamp': 1688625742.646421,
    'location': 's110',
    'vehicle_lidar_path': './PATH/T0_vehicle_lidar_robosense
       .bin',
    'vehicle_sweeps': [],
    'infrastructure_lidar_path': './PATH/T0_s110
       _lidar_ouster_south.bin',
    'infrastructure_sweeps': [],
    'registered_lidar_path': './PATH/T0_registered.bin',
    'registered_sweeps': [],
    'vehicle2infrastructure': [...],
    'lidar_anno_path': './PATH/T0_anno_registered.json',
    'vehicle_cams': {
        'vehicle_camera_basler_16mm': {
            'data_path': '.PATH/T0_vehicle_camera_basler_16
               mm.jpg',
            'type': 'vehicle_camera_basler_16mm',
            'lidar2image': array([...], dtype=float32),
            'sensor2lidar': array([...], dtype=float32),
            'camera_intrinsics': array([..], dtype=float32),
            'timestamp': 1688625742.646421}
    },
    'infrastructure_cams': {
        's110_camera_basler_south1_8mm': {
            'data_path': './PATH/T0_camera_basler_south1_8mm
               .jpg',
            'type': 's110_camera_basler_south1_8mm',
            'lidar2image': array([...], dtype=float32),
            'sensor2lidar': array([...], dtype=float32),
            'camera_intrinsics': array([..], dtype=float32),
            'timestamp': 1688625742.646421},
        's110_camera_basler_south2_8mm': {
            'data_path': './PATH/T0_s110_camera_basler_south
               2_8mm.jpg',
            'type': 's110_camera_basler_south2_8mm',
            'lidar2image': array([...], dtype=float32),
            'sensor2lidar': array([...], dtype=float32),
            'camera_intrinsics': array([..], dtype=float32),
            'timestamp': 1688625742.646421},
        's110_camera_basler_north_8mm': {
            'data_path': './PATH/T0_s110
               _camera_basler_north_8mm.jpg',
            'type': 's110_camera_basler_north_16mm',
            'lidar2image': array([...], dtype=float32),
```

```
            'sensor2lidar': array([...], dtype=float32),
            'camera_intrinsics': array([..], dtype=float32),
            'timestamp': 1688625742.646421}
    },
    'gt_boxes': array([...], dtype=float32),
    'gt_names': array([...]),
    'num_lidar_pts': array([...]),
    'num_radar_pts': array([...]),
    'valid_flag': array([...])
},
{'timestamp': 1688625743.446557, ...}
}
```

Listing 4.1: training set information .pkl file

Furthermore, a database file containing the information of each individual object in the training dataset should be created. This database information is later used by the **UnifiedObjectSampleCoop** augmentation, which is explained in Section 4.2. Creating the database involves two steps. First, a pickle file containing the information of each object should be generated, and a sample is shown in Listing 4.2. The first level keys in the pickle files are the classes (and the sample shows the 'BUS' class as the first class in the file). Then, a list containing a particular class's instance information is created for each class. Each instance here corresponds to one bounding box and has the following information:

- 'name': Name of the class
- 'path': Path to the bin file containing the exact points inside the box
- 'gt_idx': Index of that bounding box within the time stamp
- 'box3d_lidar': The LiDARInstance3DBoxes, which contains information about the location, size, and orientation of the bounding box
- 'num_points_in_gt': The number of points in the bounding box
- 'difficulty': Difficulty of identifying the box which is always set to zero.

As we observe that each instance needs a corresponding .bin file associated with it, these points are extracted from the registered point cloud file and saved in a separate .bin file.

4.1.2 Data Loader

The TUMTraf Cooperative dataset was labeled in the OpenLABEL format, which differs from the labeling convention used by the nuScenes dataset. According to the MMDetection3D standard, a custom dataloader is needed to load this dataset. As such a new dataloader is created to read the corresponding json files. There are 2 significant differences between the nuScenes data loader and the TUMTraf Cooperative data loader. The first difference is that the nuScenes dataset only contains vehicular data. As such, the keys for the images, point clouds, and sensor intrinsic and extrinsic parameters are specific to vehicular data. On the other hand, the TUMTraf intersection dataset has both infrastructure and vehicular data. In addition to having particular keys for the infrastructure and vehicular data, it also contains the transformation matrices between the two coordinate systems. As such, the data loader was modified to include all these keys and corresponding values. Another difference is the

tokens used by the nuScenes dataset to identify each instance separately. Since the TUMTraf dataset does not use tokens, these values were replaced by timestamps.

```
{
'BUS': [
    {
        'name': 'BUS',
        'path': 'gt_database/TIMESTAMP0_BUS0.bin',
        'gt_idx': 0,
        'box3d_lidar': [15.49 , ... ,  0. ],
        'num_points_in_gt': 406,
        'difficulty': 0,
        'group_id': 0
    },
    {
        'name': 'BUS',
        'path': 'gt_database/TIMESTAMP0_BUS1.bin',
        'gt_idx': 1,
        'box3d_lidar': [20.20 , ...,  0. ],
        'num_points_in_gt': 1076,
        'difficulty': 0,
        'group_id': 1
    },
    {
        'name': 'BUS',
        'path': 'gt_database/TIMESTAMP1_BUS0.bin',
        'gt_idx': 0,
        'box3d_lidar': [15.44 , ...,  0. ],
        'num_points_in_gt': 295,
        'difficulty': 0,
        'group_id': 27
    },
    ...
    ]
...
}
```

Listing 4.2: database .pkl file

4.1.3 Data preprocessing pipeline

The entire data preprocessing pipeline is shown in Figure 4.1. The blue, red, and green boxes correspond to functions that only affect the point cloud, image data, and ground truth annotations, respectively. The white boxes interact with different modalities. The first stage of the pipeline is to load the images, point clouds, and ground truth labels. These functions are performed by the following classes:

- **LoadMultiViewImageFromFilesCoop**: Load vehicular image and infrastructure images from multiple viewpoints.

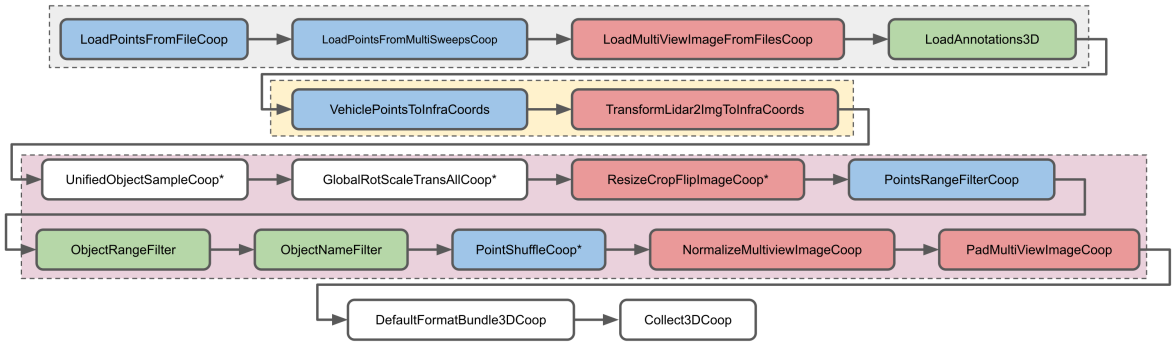

Figure 4.1: Data pipeline, including loading, augmentation, and datatype conversion stages.

- **LoadPointsFromFileCoop**: Load point cloud from vehicular and infrastructure LiDAR sensor.
- **LoadPointsFromMultiSweepsCoop**: Load points from multiple sweeps. This can be ignored for TUMTraf dataset since the LiDAR doesn't use multiple sweeps during records but is required for a dataset such as nuScenes.
- **LoadAnnotations3D**: Load the the 3D bounding box.

Once the data and labels have been loaded, all the vehicular data is converted to the infrastructure coordinate system since the labels are created from the infrastructure point of view. The vehicular point clouds are first converted in the infrastructure coordinate system in the **VehiclePointsToInfraCoords** step using the *vehicle2infrastructure* projection matrix. Next, the vehicular images must be referenced from the infrastructure point of view. This is done in the **TransformLidar2ImgToInfraCoords**, which updates the vehicular *lidar2image* projection from the vehicular perspective to the infrastructure perspective. i.e., Infrastructure LiDAR to vehicular image projection matrix. After this step, all the data is relative to the infrastructure coordinate system.

Next, the data is augmented using multiple data augmentation methods, which are explained in detail in the next section. Finally, the **DefaultFormatBundle3DCoop** step converts the data and labels from an array to a format compatible with MMDetection3D framework known as *Data Containers*, and key names of the data ("points" or "images") and metadata such as the projection matrices are added in the **Collect3DCoop** stage.

The training pipeline contains all these steps. However, the test and evaluation pipelines have specific modifications in the augmentations, which are explained in the following section. These stages in the data preprocessing pipeline are the same for all the models when using the TUMTraf cooperative dataset. However, some components can be updated/ignored when using different model configurations. For instance, if a LiDAR-only model is used, then loading the image becomes unnecessary, and the corresponding augmentations can also be dropped, and vice-versa. On the other hand, when using the TUMTraf infrastructure dataset, the vehicular data and meta information are not present. As such, this information is not loaded in the preprocessing pipeline.

4.2 Data augmentation

For robust training, the data is augmented using different data augmentation techniques. This includes augmenting both the point clouds and the images using various augmentations such

as rotation, scaling, flips, crops, and filters. This augmentation must be applied collectively for multi-modal data so that the final point cloud and image data correspond with each other.

The first augmentation is weighted object sampling, which is done in the **UnifiedObjectSampleCoop** step. The data statistic section shows that the objects are not distributed evenly. As such, to compensate for the bias in the dataset, the objects are sampled according to the ratio of their occurrences and the hardness in identifying them. The hardness values are set equally in each case since the dataset is not labeled for this purpose. However, the weights are allocated according to the ratio of the objects in the dataset as *CAR = 2, TRAILER = 5, TRUCK = 3, VAN = 3, PEDESTRIAN = 7, BUS = 5, BICYCLE = 7*. These values are selected since there are many cars, trucks, and vans in the dataset, so they're sampled at a lower ratio. In contrast, the number of buses and bicycles is comparatively lower. Furthermore, the orientation of pedestrians is challenging to predict, and thereby the ratio is set higher. The objects are sampled from the created database as explained in Section 4.1.1, and all the other labels not in the sampled data are ignored. The augmentation should also sample both the image and corresponding point cloud data.

Next, the rotation, translation, and scaling augmentations are performed. Point clouds are augmented this way in the **GlobalRotScaleTransAllCoop** step, where the point clouds are rotated, translated, and scaled by a random value. The ranges of these values are kept the same as for the original CMT implementation with the rotation range of [-0.395, 0.395] radians, the scale in a range of [0.95, 1.05], and the translation range kept at 0. Note that these augmentations should be performed the same for both the vehicular point clouds and image point clouds, and the augmentations should also be reflected in the labels and images. As such, the ground truth bounding boxes are also augmented by the same values, whereas in the case of images, the projection matrices of *lidar2image* for both the vehicle and infrastructure are updated.

Next, the images are augmented using resize, flips, and crops in the **ResizeCropFlipImageCoop** stage. The image resize value is selected randomly from a range of [0.94, 1.25], and horizontal flips are performed with a probability of 0.5. Again, these augmentations must be reflected in the transformation matrices, and as such, the projection matrices *lidar2img* and *lidar2cam* for both the infrastructure and vehicular perspectives are updated.

Next, the points in the point cloud are filtered within a particular range of interest through the **PointsRangeFilterCoop** function. In the case of the TUMTraf dataset family, this range is set to [-72m, 72m] in the X and Y axes and [-8m, 0m] in the Z axis. These ensure that anomalous points are filtered out. Similarly, the detected objects within this range are also filtered using the **ObjectRangeFilter** function based on the center points. Furthermore, certain object classes are also filtered out using the **ObjectNameFilter** function to classes with low occurrences. Finally, **PointShuffleCoop** shuffles the ordering of the points. **NormalizeMultiviewImageCoop** normalizes all the images, and **PadMultiViewImageCoop** ensures that the image sizes are a multiple of 32 for compatibility with the feature extraction backbone.

All the augmentations are used during training. However, during the testing and validation, the image and point cloud augmentations are not performed. As such, only the the **PointsRangeFilterCoop**, **ObjectRangeFilter**, **ObjectNameFilter**, **NormalizeMultiviewImageCoop**, and **PadMultiViewImageCoop** functions are used. The augmentations used only during training are specified with an asterisk(*) in Figure 4.1. Note that all these augmentations are performed for the TUMTraf intersection dataset, but some augmentations can be ignored for different model configurations. For instance, if a LiDAR-only model is used, then image augmentations are unnecessary and vice-versa.

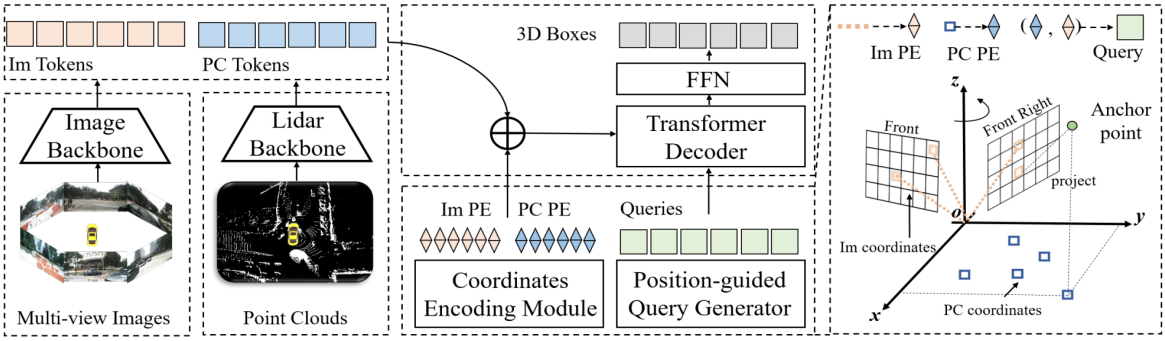

Figure 4.2: Architecture of CMT model proposed in [Yan+23a].

4.3 Model architecture

The proposed CMTCoop model is based on CMT [Yan+23a], which proposes a vehicular-viewpoint multi-modal 3D object detection model. The CMTCoop model uses a transformer-based multi-modal feature extractor similar to the one in CMT on the vehicle and infrastructure sides separately. These extracted features are then fused using a fusion module before being passed to the detection head. The architecture of the original CMT model is explained briefly first, and then the modification made for the CMTCoop model is discussed in the following subsections.

4.3.1 CMT architecture

Figure 4.2 shows the comprehensive architecture of the CMT model proposed in [Yan+23a]. First, multi-view images and LiDAR points are processed through two separate backbones to extract images and point cloud tokens. Then, the 3D coordinates of these tokens are encoded into them through a coordinate encoding module (CEM). In the meantime, queries are generated by the position-guided query generator. These queries are passed together with the position-encoded image and point tokens to the transformer decoder, subsequently predicting both the object class and the 3D bounding boxes. Further information about each of these steps can be found in the original paper.

Feature extraction backbone

Separate feature extraction backbones are used for the multi-view images and point clouds. The authors of the original work have used VoVNet and ResNet as image backbones to extract the 2D image features and VoxelNet and PointPillars as the backbone to extract the point-cloud features. However, in the context of this project, only VoVNet was used as the image extraction backbone, and VoxelNet was used as the point cloud feature extraction backbone. The feature extraction backbones thereby produce features that are referred to as tokens.

Coordinate encoding module

The coordinate encoding module (CEM) encodes that 3D position information into the tokens generated by the feature extraction backbones. Positional encodings (PE) are created for each token based on their indexing, which in the case of images corresponds to the 2D position of pixels and, in the case of point clouds, corresponds to the BEV position. These positional encodings are then used to align the individual tokens in 3D space.

Position-guided Query Generator

The position-guided query generator generates queries which are used by the transformer to generate deep fused features. First, multiple anchor points are generated within the 3D space within which the bounding boxes must be predicted. These anchor points are then projected to the image plane and the BEV plane, and the positional encoding for each anchor point is generated using the same methodology employed in the CEM.

Transformer decoder

The decoder from the original DETR implementation is used as the decoder. The generated queries in the position-guided query generator are updated in each transformer layer by interacting with the position-encoded tokens. We refer to these updated tokens as deep-fused features, and the detection head uses these deep-fused features for 3D object detection and classification.

Detection head and Loss

The detection head is made of 2 feed-forward networks (FFN), one to predict the position and orientation of the bounding box and the other to predict the object class. Once the location and classes have been predicted, they're matched with the ground truth objects through the Hungarian matching algorithm and a weighted combination of focal loss for classification and L2 loss for regressing the bounding box.

4.3.2 CMTCoop architecture

CMTCoop uses the same components as the original CMT modal, and the overall architecture of the model is shown in Figure 4.3. The major difference is that it requires separate deep feature extractors on the vehicular and infrastructure sides. These extracted deep features are then fused using a deep feature fusion model. Finally, the fused features are passed to the detection head and trained similarly to the original CMT model.

Deep feature extractor

The deep feature extractor is the multi-modal feature extractor, which lies on both the vehicle and infrastructure sides. It comprises the image and point cloud feature extraction components, the CEM, the Position-guided Query Generator, and the Transformer decoder. This module generates deep multi-modal features in both the vehicle and infrastructure sides.

Deep feature fusion neck

This component fuses the deep multi-modal features from the vehicle and infrastructure sides. We employ a simple max-pooling layer, combining each dimension's feature and considering the maximal value. For instance, each deep multi-modal feature has a dimension of H*W*L*D (which is determined by the parameters of the transformer decoder), and the final fused features would have the same dimension. Section 7.4 in the future works chapter discusses using more complicated fusion necks.

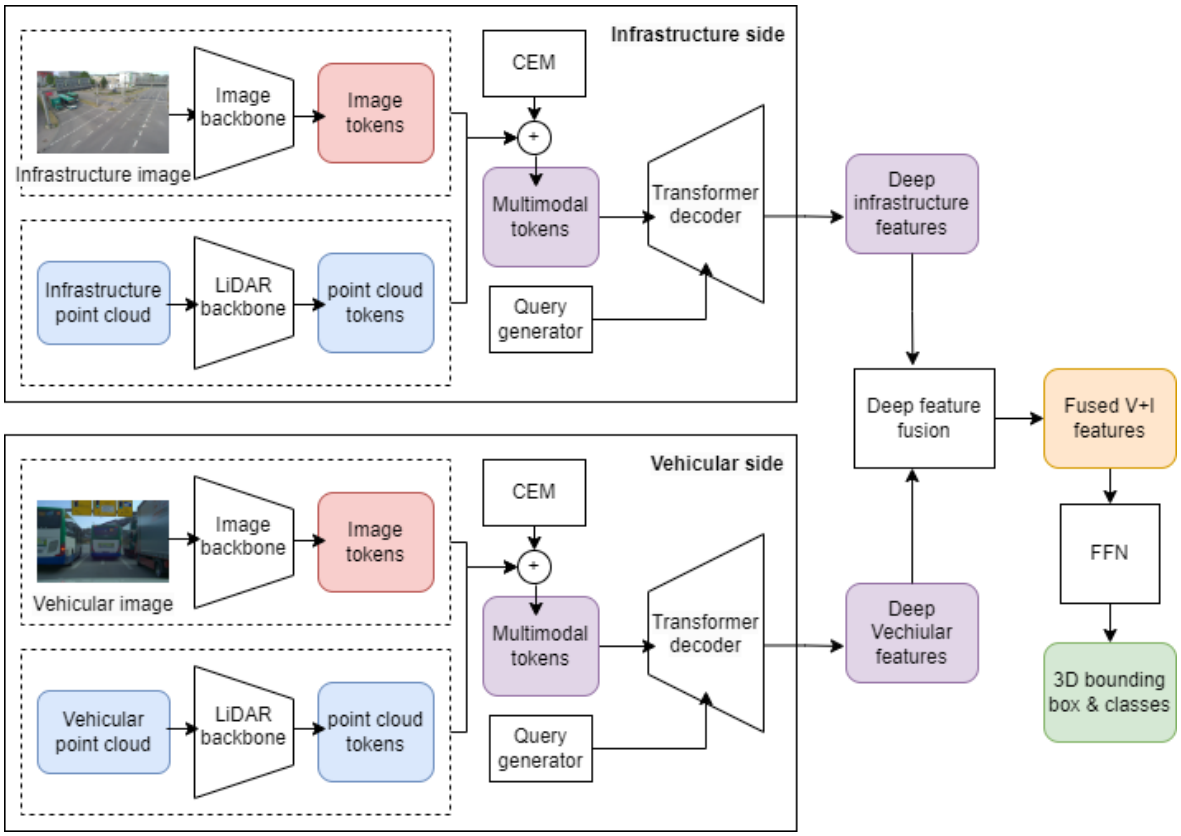

Figure 4.3: Architecture of the proposed CMTCoop model.

Cooperative detection head and Loss

The cooperative detection head and the loss are the same as the ones from the CMT model. Section 7.4 in the future works chapter discusses using other detection heads and their limitations.

4.3.3 Implementation details

In addition to the hyperparameter defined above, the model was trained with an image size of 1600x640 pixels as the final size, and 1440 x 1440 x 40 voxels. The model was trained for 20 epochs with a cosine annealing learning rate scheduler with the default values as CMT. The model was trained on a GPU cluster containing three NVIDIA RTX 3090 with 24 GB VRAM, an AMD EPYC 7282 16-Core Processor, and 32 GB RAM. Further details on the implementation can be found in the GitHub repository. [1]

[1] GitHub repository : https://github.com/suren3141/CMT-Cooperative-Perception

Chapter 5

Experiments and Ablation studies

This chapter explains the experiments carried out to benchmark the model and the ablation studies conducted to choose the best model parameters.

5.1 Evaluation Metrics

Multiple metrics were used to measure the model's efficacy, efficiency, and complexity. While efficacy plays a significant role in the accuracy of object detection, it should also be noted that these detections must occur in real-time, and as such efficiency is also of significant importance. Furthermore, the models should also be able to run on end-device. As such, the number of model parameters is limited by such constraints.

5.1.1 Model prediction accuracy

The model's efficacy is measured in terms of the mean average precision (mAP). The metric is defined in terms of two other metrics: precision and recall, which are defined as follows :

$$precision = \frac{TP}{TP + FP} \tag{5.1}$$

$$recall = \frac{TP}{TP + FN} \tag{5.2}$$

where *TP, FP* and *FN* refer to true positives, false positives, and false negatives. In object detection, precision is the proportion of correct detection out of all detection, and recall is the proportion of correct detection out of all ground truth objects. Consequently, a high precision indicates that the model makes fewer incorrect detections, and a higher recall suggests that the model detects most of the ground truth objects. Therefore, both these metrics are essential in measuring a model's efficacy, and autonomous driving recall is of higher importance since a missed detection is severe compared to an incorrect detection.

Object detection is a regression problem since both the center coordinates and dimensions are continuous values. As such, the precision of models can vary based on the error threshold used. Thus, average precision (AP) is the precision value that is calculated for different threshold values and is defined as follows:

$$AP = \text{mean}\,(precision_{\tau}) \tag{5.3}$$

where $\tau \in [.5, 1.0, 2.0, 3.0, 4.0]$ in our case is the L2 distance between the center coordinates of the predicted bounding box and the ground truth. In a multiclass object detection problem,

the AP is calculated for each class separately, and then the mean of all APs is considered. This is defined as the mean average precision (mAP).

In our experiments, two types of mAP values are considered: the BEV-mAP, which considers only the distance between the center coordinates when measuring the precision, and the 3D-mAP, which considers the 3D intersection of union (IOU) when calculating the precision.

5.1.2 Model efficiency

The model efficiency is measured in terms of the number of frames per second (FPS) processed by the model. While real-time systems require more than 24 FPS speedup, prior works [24] have highlighted that at least 10 FPS is required for near real-time systems. The FPS calculation is based on the evaluation model, and the evaluation time includes the augmentation, feature extraction, and detection steps. In addition, the first 10% of the frames (warmup stage) are ignored to avoid the time taken for initial overhead.

5.1.3 Model complexity

While cooperative object detection models can be trained on high-performance systems, it must be possible to use them on edge devices with limited resource capabilities during deployment. As such, the model complexity is often constrained in the live system by the performance of the ego vehicle system. In our case, the limiting factor is often the number of model parameters that can be fit into the edge device's GPU, which is proportional to the GPU memory usage during run-time. As such the maximum GPU memory usage during evaluation was used as the metric to measure the model complexity. In addition, the maximum GPU usage during training time was also measured as a comparative metric.

5.2 Dataset and models for benchmarking

The experiments were conducted on two different datasets, namely the TumTraf intersection dataset [Zim+23b] and the TumTraf Cooperative dataset [24]. The training set of the TUMTraf Intersection Dataset was first augmented using the infrastructure-specific augmentations in Section 4.2, and the experiments utilized the dataset's test set, which only consists of the six classes considered in [Zim+23a], namely CAR, TRUCK, PEDESTRIAN, BUS, MOTORCYCLE, and BICYCLE. On the other hand, the TumTraf cooperative dataset's test set consists of seven classes, namely CAR, TRAILER, TRUCK, VAN, PEDESTRIAN, BUS, and BICYCLE, which have a non-negligible occurrence.

The SOTA late-fusion model proposed in InfraDet3D [Zim+23a] was used as a single-viewpoint (infrastructure) benchmark on the TumTraf intersection dataset. In addition, BEVFusionCoop proposed in [24] was used as a benchmark for the TumTraf Cooperative dataset.

5.3 Ablation studies

In addition to the experiments above, different ablation studies were performed to show the reason for choices, such as transfer learning and hyperparameters. These ablation studies are explained in detail below.

5.3.1 Effect of transfer learning on model performance

Initially, the models were trained from scratch without any pretraining. In the case of LiDAR-based models for both the TUMTraf intersection dataset and cooperative dataset, it was possible to train the models to get a non-zero **mAP**. However, in the case of camera-based models, the mAP was always zero, so the VoVNet was pre-trained on the ImageNet [Lee+19] dataset. This choice was also followed in the original CMT implementation, and the pre-trained models, and the reasoning for choosing this pre-trained backbone can be found in the source.

In addition to this, the CMTCoop model was initially trained without any transfer learning (except the pre-trained VoVNet backbone). Then, various configurations of pre-trained models were used to observe the variation in the performance of the final model. Initially, only the backbones that generate the image and point cloud feature tokens were used as pre-trained models. In addition, experiments were also conducted using transfer learning on the entire model, including the transformers. Furthermore, when using transfer learning on CMTCoop, the experiments were conducted by training the original CMT model on either the TUMTraf intersection dataset or the NuScenes dataset. In addition, different components of the CMTCoop model were first pretrained from the TUMTraf cooperative dataset, and then the entire model was retrained on the TUMTraf cooperative dataset. The various configurations used for pretraining are described in the later section.

5.3.2 Performance of model on unseen data

We discussed that the proposed model was trained only using 500 day-time data samples. However, the TUMTraf cooperative dataset was later extended to include 300 more day and night data samples. In the context of this work, only the first 500 samples were used for most of the previous experiments to maintain consistency with the experiments conducted in [24]. However, another issue is overfitting since the model is trained using a dataset containing very few biased samples. As such, in this ablation study, we first observe the model's performance without any retraining on the test set of the extended dataset. Then, the model is retrained on the large dataset, including the 300 additional data samples. The model efficacy is measured in terms of the mAP values to observe the effect of overfitting.

Chapter 6

Results and Discussion

This chapter discusses the results from the experiments specified in the chapter 5 in both qualitative and quantitative manner. First, the results of the TUMTraf cooperative dataset are discussed, and next, the results of the TUMTraf intersection dataset are discussed.

6.1 TUMTraf cooperative dataset

First, the different metrics for multiple configurations of the CMTCoop model are analyzed. Then, these values are compared with the BEVFusionCoop model. Next, through qualitative analysis, specific scenarios in which the proposed model's best configuration performs better than its counterparts are observed. Finally, through the results of ablation studies, the importance of various decisions, such as transfer learning and overfitting, are depicted.

6.1.1 Performance metrics in different configurations

Table 6.1 shows all the quantitative metrics for each combination of viewpoints (Domain: vehicle/infrastructure / cooperative) and modalities (Modality: camera / LiDAR/camera + LiDAR fusion).

Table 6.1: Evaluation metrics of CMTCoop model on TUMTraf cooperative dataset.

Config.		mAP_{BEV} ↑	mAP_{3D} ↑				FPS	VRAM usage	
Domain	**Modality**		**Easy**	**Mod.**	**Hard**	**Avg.**		**Train**	**Eval**
Vehicle	Camera	69.76	68.76	79.85	66.44	69.30	13.4	5840 MiB	2645 MiB
Vehicle	LiDAR	88.17	87.94	88.53	71.99	84.72	15.0	5343 MiB	2167 MiB
Vehicle	Cam+LiDAR	91.65	84.83	91.32	72.18	85.57	8.0	9603 MiB	2759 MiB
Infra.	Camera	71.89	70.86	80.38	58.72	71.66	7.7	10528 MiB	3951 MiB
Infra.	LiDAR	94.42	91.28	95.60	77.48	91.89	17.0	5392 MiB	2175 MiB
Infra.	Cam+LiDAR	96.09	91.94	95.15	**82.35**	92.16	5.8	11535 MiB	4067 MiB
Coop.	Camera	84.07	81.03	90.05	77.94	83.43	5.6	22358 MiB	4523 MiB
Coop.	LiDAR	96.68	92.18	96.77	82.20	93.43	9.8	7352MiB	2293 MiB
Coop.	Cam+LiDAR	**97.31**	**93.70**	**96.65**	79.84	**94.10**	4.5	22468 MiB	4701 MiB

We observe that, in general, the vehicular viewpoint-only model performs the worst, followed by the infrastructure viewpoint model, and the cooperative perception model performs the best. Furthermore, the camera-only model performs much worse than LiDAR-only models, whereas camera-lidar fusion provides slight performance improvements. The comparatively higher performance of every cooperative model compared to their unimodal counter-

parts shows the importance of cooperative perception. Similarly, we also observe that fusion works better compared to single-sensor detection.

In terms of efficiency, none of the cooperative models work in near-real time (FPS > 10). However, some of the infrastructure-only and vehicle-only models provide near real-time predictions while having a relatively high **mAP** value. We also observe that most fusion models have a memory consumption of around 4 GB, and most LiDAR/camera-only models have a memory consumption of about 2 GB. As such, deploying both models on end devices with relatively good GPUs should be feasible.

Another interesting observation is comparing models that use images from vehicular perspective (row 1 and row 3) to infrastructure perspective (row 4 and row 6), the memory usage of these models increases, and the FPS sees a considerable drop. On the other hand, when considering models that use only the LiDAR information (row 2 and row 5), the memory usage and FPS are nearly equal. This occurs since the infrastructure has three cameras, whereas the vehicle has only one, resulting in a significant difference in complexity and efficiency. On the other hand, both the vehicle and infrastructure have only one LiDAR. Thus, the model complexities and efficiency measures are nearly equal. This highlights the issue of using multi-view images in object detection, as the efficiency decreases with the number of images, though the accuracy may increase.

6.1.2 Comparison of selected metrics of CMTCoop and BEVFusionCoop models

Table 6.2 shows the comparative metrics of the CMTCoop model and the BEVFusion model on the TUMTraf cooperative dataset. The BEV mAP and the average 3D mAP were chosen as measures of prediction efficacy since they express the models' ability to accurately predict the 3D center position and the bounding boxes separately. The FPS and memory consumption were also included during the evaluation to measure the models' efficiency and complexity.

Table 6.2: CMTCoop and BEVFusionCoop model evaluation metrics on TUMTraf cooperative dataset.

Config.		CMTCoop				BEVFusionCoop			
Domain	**Modality**	**mAP_{BEV} ↑**	**mAP_{3D} ↑ (Avg.)**	**FPS**	**VRAM (eval)**	**mAP_{BEV} ↑**	**mAP_{3D} ↑ (Avg.)**	**FPS**	**VRAM (eval)**
Vehicle	Camera	69.76	69.30	13.4	2645 MiB	46.83	30.36	18.42	2815 MiB
Vehicle	LiDAR	88.17	84.72	15.0	2167 MiB	85.33	80.11	55.04	2561 MiB
Vehicle	Cam+LiDAR	91.65	85.57	8.0	2759 MiB	84.90	76.40	20.5	3835 MiB
Infra.	Camera	71.89	71.66	7.7	3951 MiB	61.98	35.04	16.6	3459 MiB
Infra.	LiDAR	94.42	91.89	17.0	2175 MiB	92.86	84.88	57.76	2793 MiB
Infra.	Cam+LiDAR	96.09	92.16	5.8	4067 MiB	92.92	87.01	17.28	4585 MiB
Coop.	Camera	84.07	83.43	5.6	4523 MiB	68.94	45.74	13.9	3541 MiB
Coop.	LiDAR	96.68	93.43	9.8	2293 MiB	93.93	85.86	47.26	2857 MiB
Coop.	Cam+LiDAR	**97.31**	**94.10**	4.5	4701 MiB	94.22	90.76	11.2	4631 MiB

From table 6.2, it is evident that CMTCoop outperforms BEVFusionCoop in both the efficacy metrics (**mAP_{BEV}** & **mAP_{3D}**) in all possible combinations of domains and modalities. However, considering the efficiency, the BEVFusionCoop model performs faster than CMTCoop. This is mainly due to the many steps taken in BEVFusionCoop to improve the speedup, and incorporating this into CMTCoop is discussed in section 7.2 as future works. However, when considering the best-performing BEVFusion model (row 9) against the CMTCoop infrastructure-only LiDAR-only model (row 5), we observe that the proposed model still outperforms BEVFusionCoop in all metrics.

Figure 6.1: Vehicular (left) and Infrastructure south 2 (right) perspectives of camera.

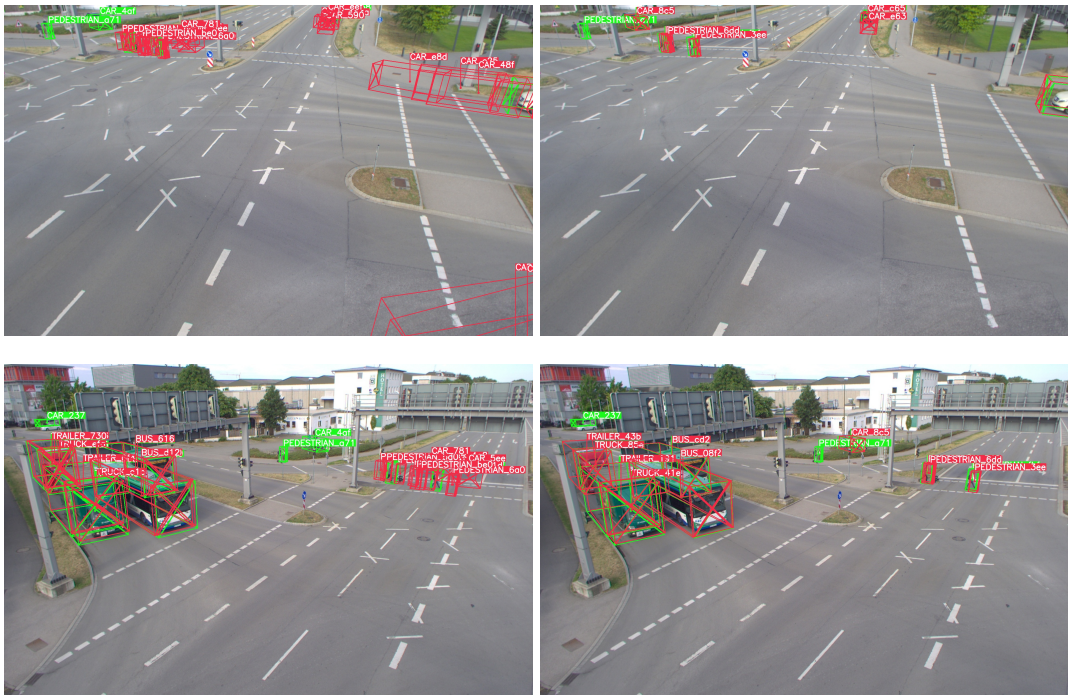

Figure 6.2: Detections from Vehicular-only (left) and Cooperative (right) models from South 1 camera (row 1) and South 2 camera (row 2) perspectives.

6.1.3 Qualitative analysis of cooperative model compared to single-viewpoint model

In this section, we qualitatively compare the detection performance of different models. First, Figure 6.1 shows the images from the vehicular perspective camera and the South 2 infrastructure camera. Most of the objects observed from the infrastructure perspective are occluded in the vehicular perspective.

Consequently, in Figure 6.2, the detection from the vehicular-only fusion model and cooperative fusion models can be observed in south 1 and south 2 perspectives. The ground truth bounding boxes are depicted in green and predictions in red. We see that there are many false positives and false negatives in the case of vehicular-only perception (left side images). However, most of these can be avoided through the cooperative fusion model.

6.1.4 Ablation study - Effect of pretraining

Different configurations were used for pretraining the model. First, Table 6.3 shows the variation of final model efficacy of the vehicle-only and infrastructure-only models when they were trained from scratch, as opposed to when the CMT model was first trained on the TUMTraf intersection dataset, and then either the image backbone or the entire transformer feature extractor was transferred to the CMTCoop model. Note that in the case of camera-only models, no transfer learning still refers to using the pre-trained VoVNet on ImageNet.

Table 6.3: Performance of camera-only and LiDAR-only CMTCoop model on TUMTraf cooperative dataset with and without transfer learning.

Config.		pre-trained component	mAP_{BEV} ↑	mAP_{3D} ↑			
Domain	**Modality**			**Easy**	**Mod.**	**Hard**	**Avg.**
Vehicle	Camera	-	56.11	53.96	61.82	51.85	55.92
		CMT - VoVNet	62.85	59.29	65.89	55.53	59.89
		CMT - VoVNet + Transformer	69.76	68.76	79.85	66.44	69.30
Vehicle	LiDAR	-	40.38	39.09	41.67	30.28	33.90
		CMT - VoxelNet	71.65	67.26	72.96	61.34	69.04
		CMT - VoxelNet + Transformer	88.17	87.94	88.53	71.99	84.72
Infra	Camera	-	57.66	55.98	58.28	52.73	55.37
		CMT - VoVNet	66.91	62.72	71.58	62.96	63.41
		CMT - VoVNet + Transformer	71.89	70.86	80.38	58.72	71.66
Infra	LiDAR	-	51.28	49.54	50.21	51.69	51.37
		CMT - VoxelNet	73.61	68.75	75.39	65.52	72.82
		CMT - VoxelNet + Transformer	94.42	91.28	95.60	77.48	91.89

We observe that, in general, the model performs better when transfer learning is used than when it's trained from scratch. In addition, we observe better performance when using the pre-trained transformer in addition to using the pre-trained feature extractors. Furthermore, unlike previous observations in Table 6.1, the camera-only models without transfer learning perform better than LiDAR-only models without transfer learning. This occurs since the camera-only models still use a pre-trained VoVNet backbone trained on ImageNet.

Note that the above studies only compare the performance of camera-only or LiDAR-only models. Next, when using camera-LiDAR fusion, one can choose between a feature extractor trained from the CMT model on the TUMTraf intersection dataset or the CMTCoop model from table 6.3. For instance, in Table 6.4, the first row represents the vehicular CMTCoop model trained on TUMTraf cooperative dataset without any pretraining. The second row represents using the pre-trained VoVNet backbone of the CMT model trained on the TUMTraf intersection and the pre-trained VoxelNet backbone of the CMT model trained on the TUMTraf intersection separately. The third row represents using the pre-trained VoVNet backbone of the best-performing CMTCoop vehicle model (row 3 in Table 6.3) and the pre-trained VoxelNet backbone of the best-performing CMTCoop vehicle model (row 6 in Table 6.3).

Again, we observe that the models without transfer learning perform the worst. Furthermore, models pre-trained on the TUMTraf cooperative dataset (row 3 and row 6) perform better compared to models trained on the TUMTraf intersection dataset (row 2 and row 5). This is due to the fact that row 3 and row 6 are both pre-trained and re-evaluated on the same dataset, whereas row 2 and row 5 are pre-trained on one dataset and then re-evaluated on a different dataset. This could also lead to overfitting and less generalization, and overcoming this issue is discussed in the chapter 7.1 as a future work.

Next, in Table 6.5, we observe the performance of the CMTCoop cooperative model when it uses individual components of CMT on the TUMTraf intersection dataset, as opposed to

Table 6.4: Performance of vehicular/infrastructure only CMTCoop fusion model on TUMTraf cooperative dataset with and without transfer learning.

Config.		pre-trained component	mAP_{BEV} ↑	mAP_{3D} ↑			
Domain	**Modality**			**Easy**	**Mod.**	**Hard**	**Avg.**
		-	49.22	45.07	51.63	40.85	46.34
Vehicle	Cam+LiDAR	CMT - VoxelNet + CMT - VoVNet	90.18	81.90	89.93	74.96	82.79
		$CMTCoop_V$ - VoxelNet + $CMTCoop_V$ - VoVNet	91.65	84.83	91.32	72.18	85.57
		-	50.31	48.37	58.51	42.64	48.37
Infra	Cam+LiDAR	CMT - VoxelNet + CMT - VoVNet	95.33	88.05	94.89	82.67	90.71
		$CMTCoop_I$ - VoxelNet + $CMTCoop_I$ - VoVNet	96.09	91.94	95.15	82.35	92.16

using the components directly used from Table 6.3 or 6.4. Each configuration is explained below by the row number:

1. refers to training the CMTCoop cooperative LiDAR-only model on the TUMTraf cooperative dataset from scratch.

2. refers to using the VoxelNet backbone of CMT trained on the TUMTraf intersection for both the vehicle and infrastructure sides of the CMTCoop model

3. refers to using the VoxelNet backbone of the best-performing CMTCoop vehicular model trained on the TUMTraf cooperative dataset (row 6 from table 6.3) and the VoxelNet backbone of the best-performing CMTCoop infrastructure model trained on the TUMTraf cooperative dataset (row 12 from table 6.3)

4. refers to training the CMTCoop cooperative camera-only model on the TUMTraf cooperative dataset from scratch

5. refers to using the VoVNet backbone of the CMT model trained on the TUMTraf intersection for both the vehicle and infrastructure sides of the CMTCoop model

6. refers to using the VovNet backbone of the best-performing CMTCoop vehicular model trained on the TUMTraf cooperative dataset (row 3 from table 6.3) and the VoVNet backbone of the best-performing CMTCoop infrastructure model trained on the TUMTraf cooperative dataset (row 9 from table 6.3)

7. refers to training the CMTCoop cooperative fusion model on the TUMTraf cooperative dataset from scratch

8. refers to using the image and LiDAR backbones of the CMT model trained on the TUMTraf intersection for both the vehicle and infrastructure sides of the CMTCoop model

9. refers to using the image and LiDAR backbones of the best-performing CMTCoop vehicular model trained on the TUMTraf cooperative dataset (row 3 from table 6.4) and the image and LiDAR backbones of the best-performing CMTCoop infrastructure model trained on the TUMTraf cooperative dataset (row 6 from table 6.4)

We see that again, the choice of model used for pretraining has an effect on the performance, and similar to Table 6.4, the models pre-trained on TUMTraf cooperative dataset perform the best.

From all these ablation studies, we observe that transfer learning generally leads to improved efficacy, which can further be improved by pretraining the transformer decoders in addition to the feature extractors. This occurs mainly because transformers generally require a large amount of data, and transfer learning helps alleviate this restriction. This also highlights the need for a larger dataset since the TUMTraf cooperative dataset used in this project

Table 6.5: Performance of CMTCoop cooperative models on TUMTraf cooperative dataset with and without transfer learning.

Config.		pre-trained component		mAP_{BEV} ↑	mAP_{3D} ↑			
Domain	**Modality**	Vehicle side	Infra. side		**Easy**	**Mod.**	**Hard**	**Avg.**
		-	-	61.92	54.75	64.16	50.27	58.16
Coop.	LiDAR	CMT - VoxelNet	CMT - VoxelNet	95.06	85.27	92.96	80.58	89.52
		CMTCoop - VoxelNet	CMTCoop - VoxelNet	96.68	92.18	96.77	82.20	93.43
		-	-	72.71	68.42	75.48	55.37	69.67
Coop.	Camera	CMT - VoVNet	CMT - VoVNet	80.43	75.28	88.16	72.76	80.93
		CMTCoop - VoVNet	CMTCoop - VoVNet	84.07	81.03	90.05	77.94	83.43
		-	-	72.95	71.03	78.16	60.59	71.24
Coop.	Cam+LiDAR	CMT - VoxelNet + VoVNet	CMT - VoxelNet + VoVNet	95.63	87.85	95.19	83.75	91.06
		CMTCoop - VoxelNet + VoVNet	CMTCoop - VoxelNet + VoVNet	97.31	93.70	96.65	79.84	94.10

is comparatively small. Note that none of these studies compare the FPS or the memory usage of the models. Since the model architectures do not change, the speed or the complexity of the model also do not change, and as such, the values are the same with or without transfer learning.

6.1.5 Ablation study - Performance of model on unseen data

Table 6.6 shows the model's performance on an unseen dataset containing 800 day and night samples in this ablation study. For each pair of rows, the first row shows the model performance when trained on the training set of the initial 500 sample version of the TUMTraf cooperative dataset and then tested on the test set of the latest 800 sample version of the TUMTraf cooperative dataset. The next row shows the model performance when trained and tested on the latest 800 sample version of the dataset.

Table 6.6: Performance of CMTCoop cooperative models on the 800 extended TUMTraf cooperative dataset.

Config.		Trained on	mAP_{BEV} ↑	mAP_{3D} ↑			
Domain	**Modality**			**Easy**	**Mod.**	**Hard**	**Avg.**
Coop.	LiDAR	500 samples (day)	70.35	64.97	68.15	50.67	66.32
		800 samples (day & night)	83.42	81.13	83.81	72.45	81.98
Coop.	Camera	500 samples (day)	58.12	53.27	60.17	40.13	54.96
		800 samples (day & night)	63.94	60.28	69.67	51.56	60.36
Coop.	Cam+LiDAR	500 samples (day)	71.33	68.12	75.23	58.19	69.45
		800 samples (day & night)	85.25	80.76	85.19	70.37	82.65

The model performance decreases significantly after unseen data is introduced into the test set. However, when it's retrained on the entire dataset, the performance improves again, but this metric value is still less than the corresponding values from Table 6.1. Furthermore, figure 6.3 shows two similar situations at daytime and nighttime, and the detections of the cooperative fusion model are shown. The model makes more misdetections and misclassifications in the nighttime compared to the daytime. This indicates that the initial model has been ovetrfit to this particular dataset. As such, more data is required to generalize the model performance for different scenarios and make the model more robust.

6.2 TUMTraf intersection dataset

Table 6.7 shows the comparative 3D mAP values of the CMTCoop model with the SOTA model proposed in [Zim+23a] for the infrastructure viewpoint object detection model on the TUM-

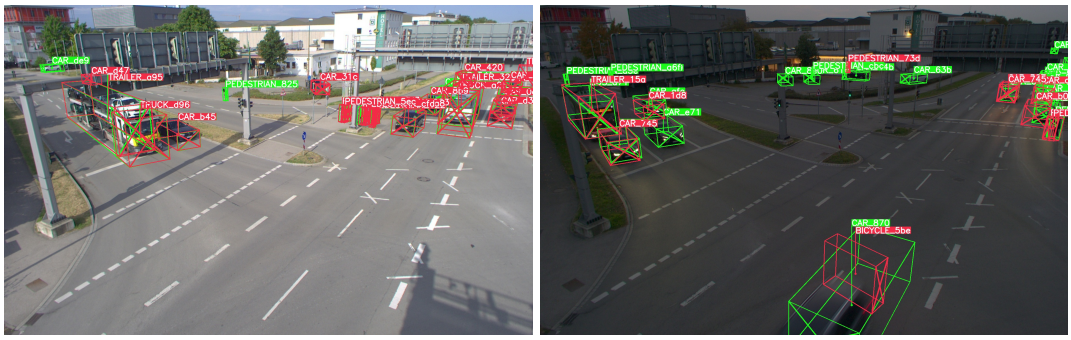

Figure 6.3: Detections of cooperative fusion model at daytime (left) and nighttime (right) from the 800 sample dataset.

Traf intersection dataset. In addition, the BEVFusion-Cooperative model proposed in [24] is also considered for comparison. The values for the InfraDet3D model and BEVFusion-Cooperative models are extracted directly from the source, whereas the TUMTraf devkit was used to obtain the values for the proposed CMTCoop model. The table shows that the proposed CMTCoop model outperforms all the other models on the TUMTraf intersection dataset.

Table 6.7: Evaluation results of infrastructure-only models on TUMTraf Intersection test set.

			$\mathbf{mAP_{3D}} \uparrow$			
Model	**FOV**[1]	**Modality**	**Easy**	**Mod.**	**Hard**	**Avg.**
InfraDet3D	south 1	LiDAR	75.81	47.66	42.16	55.21
BEVFusionCoop	south 1	LiDAR	76.24	48.23	35.19	69.47
CMTCoop	south 1	LiDAR	**80.62**	**64.46**	**50.41**	**72.68**
InfraDet3D	south 2	LiDAR	38.92	46.60	43.86	43.13
BEVFusionCoop	south 2	LiDAR	74.97	55.55	39.96	69.94
CMTCoop	south 2	LiDAR	**79.34**	**60.81**	**45.53**	**70.31**
InfraDet3D	south 1	Cam+LiDAR	67.08	31.38	35.17	44.55
BEVFusionCoop	south 1	Cam+LiDAR	75.68	45.63	**45.63**	66.75
CMTCoop	south 1	Cam+LiDAR	**80.86**	**61.37**	45.32	**70.65**
InfraDet3D	south 2	Cam+LiDAR	58.38	19.73	33.08	37.06
BEVFusionCoop	south 2	Cam+LiDAR	74.73	**53.46**	**41.96**	66.89
CMTCoop	south 2	Cam+LiDAR	**78.92**	52.67	39.76	**67.21**

Chapter 7

Future work

This chapter discusses the limitations of the proposed methodology and improvements that can be made in the future, which could lead to better performance. This includes improvements in the dataset, the pre-processing stages, and the model architecture.

7.1 Extending dataset

The proposed CMTCoop model is trained using the TUMTraf intersection and the TUMTraf cooperative datasets. The TUMTraf cooperative dataset only contains 500 labeled frames, and such few data instances are often a limitation when training complex deep learning models. Though transfer learning leads to a significant performance improvement, the limited amount of data still raises questions over the validity of the evaluation metrics, as a percentage increase in performance may correspond to a difference of a few objects identified/misidentified in the dataset. As such, the dataset should be extended to include more instances and frames.

Furthermore, the current dataset contains mostly daytime data and a few instances (100 frames) of nighttime data. This also acts as a limitation in analyzing the robustness of the model since it might have been overfitting for a particular scenario. Furthermore, specific classes have restricted movement within a particular area when observing the BEV movement patterns. For instance, pedestrian movement is limited to certain parts of the roads and crossings and a similar constraint is observed for bicycles. This further creates a class-wise bias in object distribution. As such, the model cannot be guaranteed to provide similar performance in a different set of cooperative perception data. As such, increasing the variability of the data in terms of varying weather conditions, movement patterns, and object types is another vital improvement required in the dataset.

7.2 Improving model efficiency

One of the significant issues of the proposed model over BEVFusionCoop is that the proposed model has a lower FPS in comparison. This must be improved for it to have a near-real-time performance. This can be done by updating different components of the model. By updating the voxelization step, model backbones, and feature fusion component, BEVFusionCoop has shown that evaluation becomes faster.

The proposed CMTCoop model uses SpConv for voxelization, whereas BEVFusionCoop uses torchsparse. Torchsparse is lighter and faster compared to SpConv, and as such, this

must be replaced in the future. Furthermore, the proposed model uses VoxelNet backbone for point cloud feature extraction and VoVNet for image feature extraction. In comparison, BEVFusionCoop uses PointPillars for point cloud feature extraction and Yolov8 for image feature extraction. These backbones are newer, faster, and lightweight compared to the ones used in the proposed CMTCoop model. Thereby, replacing the feature extraction backbones could lead to improved efficiency and lower model complexity.

7.3 Live system performance

This project aims to show that cooperative perception leads to better efficacy when compared to the single-viewpoint (especially vehicular) perception models. However, other issues, such as errors in projection matrices, synchronization errors, transmission delays, and V2I data compression, have been ignored for brevity. In a real-life system, these considerations play a significant role in the model's performance and, as such, cannot be dismissed. Specifically, we assume the deep features extracted from the infrastructure side can be transmitted to the vehicular side without delays or data loss/compression. However, the deep features arrive at the vehicular side with inevitable delays and losses, and prior works have attempted to rectify this. Such approaches must be incorporated into the model to overcome the issue of transmission. Another issue is the existence of synchronization errors and errors in estimating the 3D position of the vehicle (from the GPS/IMU sensor unit). These errors would propagate into the prediction and, as such, must be accounted for. Prior works have tackled this issue using uncertainty estimation approaches, and similar techniques must be included to deploy this in a real-world scenario.

7.4 Modifying components of the model

In Section 7.2, the possibility of improving the backbones was discussed to improve the model efficiency. Similarly, it would also be beneficial to experiment with different deep-feature fusion modules and detection heads. The proposed model uses a max-pooling layer as the fusion module and FFN as the detection head. In addition to these modules, further experiments were carried out with a convolution-based fusion module and detection head. However, these models couldn't be adequately trained to obtain a high mAP value. This occurs mainly due to the lack of data with higher model complexities. As such, once the dataset has been extended, it would be worthwhile to carry out different experiments by changing the fusion module and detection head to improve the model performance.

Furthermore, overlapping detections were another issue often observed in the proposed model, and the metrics considered in this work disregard this issue. However, in practice, overlapping prediction boxes must be removed. Prior works have attempted this through 3D nonmaximum suppression (3D-NMS), which should be incorporated in future works.

Chapter 8

Conclusion

This project proposes CMTCoop - a cooperative perception model for 3D object detection on cooperative V2I perception data. The proposed CMTCoop model improves the efficacy compared to single viewpoint models and leads to a +8.53 improvement in 3D mAP compared to vehicular-only perception. Furthermore, this work also shows that multi-modal sensor fusion leads to better performance over single-modality-based object detection and further clarifies these observations through qualitative analysis. The proposed model also improves the mAP on the TUMTraf cooperative dataset over the SOTA model - BEVFusionCoop. Furthermore, the proposed deep fusion model performs better than the late fusion model - InfraDet3D on the TUMTraf intersection dataset.

Though the efficiency is comparatively low for the best-performing model, other model configurations provide near-real-time run time while providing adequate mAP performance. The work also shows the importance of transfer learning for the proposed transformer-based model. Through ablation studies, we discuss the difference in model performance for different pre-trained backbones and hyperparameters. Finally, the work also highlights the shortcomings of the proposed model and discusses steps that can be taken in the future to improve the performance and avoid issues such as overfitting. Also, considering brevity, transmission latency, and sensor synchronization have been disregarded, and these should also be discussed in future works in the proposed solution.

Bibliography

[Bai+22] Bai, Z., Wu, G., Barth, M. J., Liu, Y., Sisbot, E. A., and Oguchi, K. "Pillargrid: Deep learning-based cooperative perception for 3d object detection from onboard-roadside lidar". In: *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE. 2022, pp. 1743–1749.

[Bai+23] Bai, Z., Wu, G., Barth, M. J., Liu, Y., Sisbot, E. A., and Oguchi, K. "Vinet: Lightweight, scalable, and heterogeneous cooperative perception for 3d object detection". In: *Mechanical Systems and Signal Processing* 204 (2023), p. 110723.

[BL19] Brazil, G. and Liu, X. "M3d-rpn: Monocular 3d region proposal network for object detection". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 9287–9296.

[Bus+22] Busch, S., Koetsier, C., Axmann, J., and Brenner, C. "LUMPI: The Leibniz University Multi-Perspective Intersection Dataset". In: *2022 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2022, pp. 1127–1134.

[Cae+20] Caesar, H., Bankiti, V., Lang, A. H., Vora, S., Liong, V. E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., and Beijbom, O. "nuscenes: A multimodal dataset for autonomous driving". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 11621–11631.

[Car+20] Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., and Zagoruyko, S. "End-to-end object detection with transformers". In: *European conference on computer vision*. Springer. 2020, pp. 213–229.

[Con20] Contributors, M. *MMDetection3D: OpenMMLab next-generation platform for general 3D object detection*. https://github.com/open-mmlab/mmdetection3d. 2020.

[Cre+22] Creß, C., Zimmer, W., Strand, L., Fortkord, M., Dai, S., Lakshminarasimhan, V., and Knoll, A. "A9-Dataset: Multi-Sensor Infrastructure-Based Dataset for Mobility Research". In: *2022 IEEE Intelligent Vehicles Symposium (IV)*. 2022, pp. 965–970. DOI: 10.1109/IV51971.2022.9827401.

[Dol+22] Doll, S., Schulz, R., Schneider, L., Benzin, V., Enzweiler, M., and Lensch, H. P. "Spatialdetr: Robust scalable transformer-based 3d object detection from multi-view camera images with global cross-sensor attention". In: *European Conference on Computer Vision*. Springer. 2022, pp. 230–245.

[Dos+17] Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., and Koltun, V. "CARLA: An Open Urban Driving Simulator". In: *Proceedings of the 1st Annual Conference on Robot Learning*. 2017, pp. 1–16.

[Fan+21] Fan, L., Xiong, X., Wang, F., Wang, N., and Zhang, Z. "Rangedet: In defense of range view for lidar-based 3d object detection". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, pp. 2918–2927.

[Fan+23] Fan, S., Yu, H., Yang, W., Yuan, J., and Nie, Z. “Quest: Query stream for vehicle-infrastructure cooperative perception”. In: *arXiv preprint arXiv:2308.01804* (2023).

[Gei+13] Geiger, A., Lenz, P., Stiller, C., and Urtasun, R. “Vision meets robotics: The kitti dataset”. In: *The International Journal of Robotics Research* 32.11 (2013), pp. 1231–1237.

[He+21] He, Y., Ma, L., Jiang, Z., Tang, Y., and Xing, G. “VI-eye: Semantic-based 3D point cloud registration for infrastructure-assisted autonomous driving”. In: *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*. 2021, pp. 573–586.

[Hu+23] Hu, Y., Lu, Y., Xu, R., Xie, W., Chen, S., and Wang, Y. “Collaboration Helps Camera Overtake LiDAR in 3D Detection”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 9243–9252.

[Hua+21] Huang, J., Huang, G., Zhu, Z., Ye, Y., and Du, D. “Bevdet: High-performance multi-camera 3d object detection in bird-eye-view”. In: *arXiv preprint arXiv:2112.11790* (2021).

[Hua+23] Huang, T., Liu, J., Zhou, X., Nguyen, D. C., Azghadi, M. R., Xia, Y., Han, Q.-L., and Sun, S. “V2X Cooperative Perception for Autonomous Driving: Recent Advances and Challenges”. In: *arXiv preprint arXiv:2310.03525* (2023).

[Kum+22] Kumar, A., Brazil, G., Corona, E., Parchami, A., and Liu, X. “Deviant: Depth equivariant network for monocular 3d object detection”. In: *European Conference on Computer Vision*. Springer. 2022, pp. 664–683.

[Lan+19] Lang, A. H., Vora, S., Caesar, H., Zhou, L., Yang, J., and Beijbom, O. “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.

[Lee+19] Lee, Y., Hwang, J.-w., Lee, S., Bae, Y., and Park, J. “An Energy and GPU-Computation Efficient Backbone Network for Real-Time Object Detection”. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2019, pp. 752–760. DOI: 10.1109/CVPRW.2019.00103.

[Li+22a] Li, Y., Ma, D., An, Z., Wang, Z., Zhong, Y., Chen, S., and Feng, C. “V2X-Sim: Multi-agent collaborative perception dataset and benchmark for autonomous driving”. In: *IEEE Robotics and Automation Letters* 7.4 (2022), pp. 10914–10921.

[Li+23] Li, Y., Ge, Z., Yu, G., Yang, J., Wang, Z., Shi, Y., Sun, J., and Li, Z. “Bevdepth: Acquisition of reliable depth for multi-view 3d object detection”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 37. 2. 2023, pp. 1477–1485.

[Li+22b] Li, Z., Wang, W., Li, H., Xie, E., Sima, C., Lu, T., Qiao, Y., and Dai, J. “Bevformer: Learning bird’s-eye-view representation from multi-camera images via spatiotemporal transformers”. In: *European conference on computer vision*. Springer. 2022, pp. 1–18.

[Liu+22] Liu, Y., Wang, T., Zhang, X., and Sun, J. “Petr: Position embedding transformation for multi-view 3d object detection”. In: *European Conference on Computer Vision*. Springer. 2022, pp. 531–548.

[Liu+23] Liu, Z., Tang, H., Amini, A., Yang, X., Mao, H., Rus, D. L., and Han, S. "Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation". In: *2023 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2023, pp. 2774–2781.

[Lop+18] Lopez, P. A., Behrisch, M., Bieker-Walz, L., Erdmann, J., Flötteröd, Y.-P., Hilbrich, R., Lücken, L., Rummel, J., Wagner, P., and Wießner, E. "Microscopic Traffic Simulation using SUMO". In: *The 21st IEEE International Conference on Intelligent Transportation Systems*. IEEE, 2018. URL: https://elib.dlr.de/124092/.

[PF20] Philion, J. and Fidler, S. "Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d". In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16*. Springer. 2020, pp. 194–210.

[Qi+17] Qi, C. R., Su, H., Mo, K., and Guibas, L. J. "Pointnet: Deep learning on point sets for 3d classification and segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 652–660.

[QZ23] Qiao, D. and Zulkernine, F. "CoBEVFusion: Cooperative Perception with LiDAR-Camera Bird's-Eye View Fusion". In: *arXiv preprint arXiv:2310.06008* (2023).

[Sim+19] Simonelli, A., Bulo, S. R., Porzi, L., López-Antequera, M., and Kontschieder, P. "Disentangling monocular 3d object detection". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 1991–1999.

[Sun+20] Sun, P., Kretzschmar, H., Dotiwalla, X., Chouard, A., Patnaik, V., Tsui, P., Guo, J., Zhou, Y., Chai, Y., Caine, B., et al. "Scalability in perception for autonomous driving: Waymo open dataset". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 2446–2454.

[Sun+21] Sun, P., Wang, W., Chai, Y., Elsayed, G., Bewley, A., Zhang, X., Sminchisescu, C., and Anguelov, D. "Rsn: Range sparse net for efficient, accurate lidar 3d object detection". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 5725–5734.

[Tea23] Team, T. T. D. *TUMTraf Dataset Devkit*. 2023. URL: https://github.com/tum-traffic-dataset/tum-traffic-dataset-dev-kit (visited on 10/01/2023).

[24] "TraffiX - A V2X Dataset for Multi-Modal Cooperative 3D Object Detection of Traffic Participants Using Onboard and Roadside Sensors". In: *Submitted for CVPR 2024* (2024).

[Vor+20] Vora, S., Lang, A. H., Helou, B., and Beijbom, O. "Pointpainting: Sequential fusion for 3d object detection". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 4604–4612.

[Wan+22a] Wang, H., Zhang, X., Li, Z., Li, J., Wang, K., Lei, Z., and Haibing, R. "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.

[Wan+21] Wang, T., Zhu, X., Pang, J., and Lin, D. "Fcos3d: Fully convolutional one-stage monocular 3d object detection". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 913–922.

[Wan+20] Wang, Y., Fathi, A., Kundu, A., Ross, D. A., Pantofaru, C., Funkhouser, T., and Solomon, J. "Pillar-based object detection for autonomous driving". In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXII 16*. Springer. 2020, pp. 18–34.

[Wan+22b] Wang, Y., Guizilini, V. C., Zhang, T., Wang, Y., Zhao, H., and Solomon, J. "Detr3d: 3d object detection from multi-view images via 3d-to-2d queries". In: *Conference on Robot Learning*. PMLR. 2022, pp. 180–191.

[Wei+23] Wei, S., Wei, Y., Hu, Y., Lu, Y., Zhong, Y., Chen, S., and Zhang, Y. "Robust Asynchronous Collaborative 3D Detection via Bird's Eye View Flow". In: *arXiv preprint arXiv:2309.16940* (2023).

[Wu+23] Wu, Z., Gan, Y., Wang, L., Chen, G., and Pu, J. "MonoPGC: Monocular 3D Object Detection with Pixel Geometry Contexts". In: *arXiv preprint arXiv:2302.10549* (2023).

[Xu+22a] Xu, R., Tu, Z., Xiang, H., Shao, W., Zhou, B., and Ma, J. "CoBEVT: Cooperative bird's eye view semantic segmentation with sparse transformers". In: *arXiv preprint arXiv:2207.02202* (2022).

[Xu+23] Xu, R., Xia, X., Li, J., Li, H., Zhang, S., Tu, Z., Meng, Z., Xiang, H., Dong, X., Song, R., et al. "V2v4real: A real-world large-scale dataset for vehicle-to-vehicle cooperative perception". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 13712–13722.

[Xu+22b] Xu, R., Xiang, H., Tu, Z., Xia, X., Yang, M.-H., and Ma, J. "V2x-vit: Vehicle-to-everything cooperative perception with vision transformer". In: *European conference on computer vision*. Springer. 2022, pp. 107–124.

[Xu+22c] Xu, R., Xiang, H., Xia, X., Han, X., Li, J., and Ma, J. "Opv2v: An open benchmark dataset and fusion pipeline for perception with vehicle-to-vehicle communication". In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 2583–2589.

[Yan+23a] Yan, J., Liu, Y., Sun, J., Jia, F., Li, S., Wang, T., and Zhang, X. "Cross modal transformer: Towards fast and robust 3d object detection". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 18268–18278.

[YML18] Yan, Y., Mao, Y., and Li, B. "Second: Sparsely embedded convolutional detection". In: *Sensors* 18.10 (2018), p. 3337.

[Yan+23b] Yang, L., Yu, K., Tang, T., Li, J., Yuan, K., Wang, L., Zhang, X., and Chen, P. "BEVHeight: A Robust Framework for Vision-based Roadside 3D Object Detection". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 21611–21620.

[Ye+22] Ye, X., Shu, M., Li, H., Shi, Y., Li, Y., Wang, G., Tan, X., and Ding, E. "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.

[Yu+22] Yu, H., Luo, Y., Shu, M., Huo, Y., Yang, Z., Shi, Y., Guo, Z., Li, H., Hu, X., Yuan, J., 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.

[Yu+23a] Yu, H., Tang, Y., Xie, E., Mao, J., Yuan, J., Luo, P., and Nie, Z. "Vehicle-infrastructure cooperative 3d object detection via feature flow prediction". In: *arXiv preprint arXiv:2303.10552* (2023).

[Yu+23b] Yu, H., Yang, W., Ruan, H., Yang, Z., Tang, Y., Gao, X., Hao, X., Shi, Y., Pan, Y., Sun, N., et al. "V2X-Seq: A Large-Scale Sequential Dataset for Vehicle-Infrastructure Cooperative Perception and Forecasting". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 5486–5495.

[ZT18] Zhou, Y. and Tuzel, O. "Voxelnet: End-to-end learning for point cloud based 3d object detection". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 4490–4499.

[Zim+23a] Zimmer, W., Birkner, J., Brucker, M., Nguyen, H. T., Petrovski, S., Wang, B., and Knoll, A. C. "Infradet3d: Multi-modal 3d object detection based on roadside infrastructure camera and lidar sensors". In: *arXiv preprint arXiv:2305.00314* (2023).

[Zim+23b] Zimmer, W., Creß, C., Nguyen, H. T., and Knoll, A. C. "A9 Intersection Dataset: All You Need for Urban 3D Camera-LiDAR Roadside Perception". In: *arXiv preprint arXiv:2306.09266* (2023).

[Zim+23c] Zimmer, W., Creß, C., Nguyen, H. T., and Knoll, A. C. "TUMTraf Intersection Dataset: All You Need for Urban 3D Camera-LiDAR Roadside Perception". In: *2023 IEEE Intelligent Transportation Systems (ITSC)*. IEEE, 2023.

[ZRT19] Zimmer, W., Rangesh, A., and Trivedi, M. "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.

[Zim+23d] Zimmer, W., Wu, J., Zhou, X., and Knoll, A. C. "Real-time and robust 3d object detection with roadside lidars". In: *Proceedings of the 12th International Scientific Conference on Mobility and Transport: Mobility Innovations for Growing Megacities*. Springer. 2023, pp. 199–219.