# COOPERNAUT: End-to-End Driving with Cooperative Perception for Networked Vehicles

Jiaxun Cui*[1] Hang Qiu*[2] Dian Chen[1] Peter Stone[1,3] Yuke Zhu[1]
[1]The University of Texas at Austin [2]Stanford University [3]Sony AI
cuijiaxun@utexas.edu, hangqiu@stanford.edu, {dchen, pstone, yukez}@cs.utexas.edu

## Abstract

*Optical sensors and learning algorithms for autonomous vehicles have dramatically advanced in the past few years. Nonetheless, the reliability of today's autonomous vehicles is hindered by the limited line-of-sight sensing capability and the brittleness of data-driven methods in handling extreme situations. With recent developments of telecommunication technologies, cooperative perception with vehicle-to-vehicle communications has become a promising paradigm to enhance autonomous driving in dangerous or emergency situations. We introduce* COOPERNAUT*, an end-to-end learning model that uses cross-vehicle perception for vision-based cooperative driving. Our model encodes LiDAR information into compact point-based representations that can be transmitted as messages between vehicles via realistic wireless channels. To evaluate our model, we develop* AUTOCASTSIM*, a network-augmented driving simulation framework with example accident-prone scenarios. Our experiments on* AUTOCASTSIM *suggest that our cooperative perception driving models lead to a 40% improvement in average success rate over egocentric driving models in these challenging driving situations and a 5× smaller bandwidth requirement than prior work V2VNet.* COOPERNAUT *and* AUTOCASTSIM *are available at https://ut-austin-rpl.github.io/Coopernaut/.*

## 1. Introduction

The widespread deployment of autonomous driving and advanced driver assistance systems is challenged by safety concerns. While deep learning has improved autonomy stacks with data-driven techniques [9, 10, 42], learning-based driving policies to date are still brittle, especially in the face of extreme situations and corner cases that one might encounter only a few times every million miles of driving [8]. The lack of robustness of learning algorithms is exacerbated by the limited sensing capabilities of optical sensors on individual vehicles, such as stereo cameras and LiDAR, that are confined to line-of-sight sensing and unreliable in bad weather conditions. With the advent of new telecommunication technologies, such as 5G networks and vehicle-to-vehicle (V2V) communications, *cooperative perception* [6, 13, 21, 32] is becoming a promising paradigm that enables sensor information to be shared between vehicles (and roadside devices [45]) in real-time. The shared information can augment the field of view of individual vehicles and convey the intents and path plans of nearby vehicles, offering the potential to improve driving safety, particularly in accident-prone scenarios.

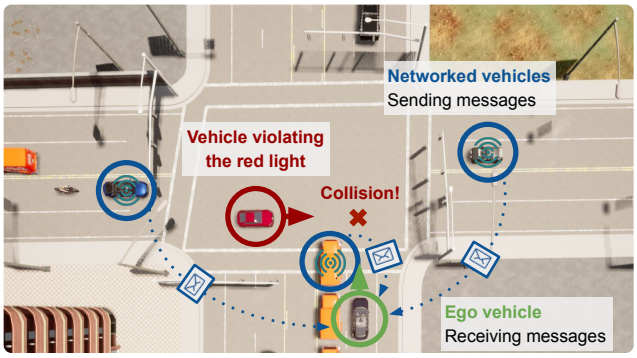


Figure 1. COOPERNAUT enables vehicles to communicate critical information beyond occlusion and sensing range for vision-based driving. The blue dashed arrows are information sharing flows. Through cooperative perception, COOPERNAUT makes more informed driving decisions when line-of-sight sensing is limited.

Ideally, learning autonomous driving policies with cooperative perception should take advantage of existing deep learning methods customized for ego perception [11, 16, 29] by considering the combined sensory data from all vehicles as an *augmented* version of on-board sensing. In practice, the efficacy of cooperative perception hinges on *what* data to transmit within the limited network bandwidth and *how* to use the aggregated information to build a coher-

* Equal contribution. Correspondence: cuijiaxun@utexas.edu, hangqiu@stanford.edu, yukez@cs.texas.edu

ent and accurate understanding of traffic situations. Recent work on cooperative driving has demonstrated the benefit of cross-vehicle perception for augmenting sensing capabilities and driving decisions [6, 21]. Nonetheless, these methods have abstracted away raw sensory data with low-dimensional meta-data. Prior work introduced 3D sensor fusion (AVR [32], Cooper [13]) and representation fusion (V2VNet [41]) algorithms that aggregate perception results from nearby vehicles via V2V channels. They focused on 3D detection and motion forecasting on static datasets, rather than interactive driving policies.

We introduce COOPERNAUT, an end-to-end cooperative driving model for networked vehicles. COOPERNAUT learns to fuse encoded LiDAR information shared by nearby vehicles under realistic V2V channel capacity. To communicate meaningful scene information from nearby vehicles while conforming to bandwidth limits, we design our driving policy architecture based on the Point Transformer [46], a self-attention network for point cloud processing. This architecture pre-processes the raw point cloud, on each networked vehicle locally, into spatial-aware neural representations. These representations are compact, which can be efficiently transmitted over realistic wireless channels. Meanwhile, they are physically grounded, thus can be spatially transformed and aggregated with ego representations. The entire architecture is end-to-end differentiable, permitting control supervision (imitating an oracle planner with access to privileged information) to flow back to the perception stack, thus ensuring the learned representations and messages contain task-relevant information.

To examine the effectiveness of COOPERNAUT, we develop a CARLA-based simulation framework, AUTOCASTSIM, where we designed three accident-prone scenarios. All the scenarios are designed to be challenging for ego perception to fully comprehend the traffic situation. AUTOCASTSIM has a built-in networking simulation for customizable multi-vehicle communications and an expert driving model with privileged information. We evaluate COOPERNAUT with voxel-based baselines [41] and different sensor fusion schemes.

In summary, our main contributions are as follows:

- We introduce COOPERNAUT, an end-to-end driving model with cooperative perception via V2V channels. Our model learns compact representations for communication that can be easily harnessed by the ego vehicle to improve its driving decisions.
- We develop a network-augmented autonomous driving simulation framework AUTOCASTSIM to evaluate COOPERNAUT and baselines in accident-prone scenarios and to promote future research on vision-based cooperative perception.
- Our results show that COOPERNAUT substantially reduces safety hazards for line-of-sight sensing. Its design improves both driving performance and communication efficiency over baselines.

## 2. Related Work

**Deep Learning for Driving Policy.** Learning a driving controller involves training closed-loop policies using deep networks, usually via imitation learning and/or reinforcement learning. Imitation learning for autonomous driving was pioneered by Pomerleau [27], and has since then been extended to urban and more complex scenarios [7, 11, 14, 15, 28, 36]. Very recently, reinforcement learning has also made progress in autonomous driving [12, 39], showing potential to train better policies in complex situations [12, 39]. However, reinforcement learning is known to be more data-hungry and requires engineering a high-quality reward function. We follow the imitation learning paradigm but use an expert oracle with complete global information [11] for training efficiency.

**3D Perception for Autonomous Driving.** 3D perception has become more popular in autonomous driving due to the decreasing cost of commoditized LiDAR sensors. Zhou and Tuzel [47] pioneered using 3D object detection in autonomous driving, and since then, it has been further developed as better models, and more advanced techniques have been discovered. Very recently, Prakash et al. [28] also explored end-to-end driving using point cloud data. Two families of 3D perception backbones have been widely adopted: *voxel*-based methods discretize points to voxels [22, 37, 47]; and *point*-based methods directly operate on coordinates [30, 31, 46]. COOPERNAUT uses a transformer-based architecture [40] with point-based representations [30, 31, 46], which preserves high spatial resolutions with discretization and requires lower bandwidths to transmit without compression needed by prior work [41].

**Networked Vehicles and Cooperative Perception.** Network connectivity offers a great potential for improving the safety and reliability of self-driving cars. Vehicles can now share surrounding information via Vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2X) channels using wireless technologies, such as Dedicated Short Range Communication (DSRC) [20] and cellular-assisted V2X (C-V2X) [18, 34]. These V2V/V2X communication devices are increasingly deployed in current and upcoming vehicle models [3, 4]). The academic community has built city-scale wireless research platforms (COSMOS [1]) and large connected vehicle testbeds (*e.g.*, MCity [5], DRIVE C2X [38]), to explore the feasibility of cooperative vehicles and applications. Prior work [13, 32] proposed cooperative perception systems that broaden the vehicle's visual horizon by sharing raw visual information with other nearby vehicles. Such systems can be scaled up to dense traffic scenarios leveraging edge servers [45] or in an ad-hoc fashion [33]. Recent work [24, 41, 43] proposed multi-

agent perception models to process sensor information and share compact representations within a local traffic network. In contrast, we focus on cooperative driving of networked vehicles with onboard visual data and realistic networking conditions, advancing towards real-world V2V settings.

## 3. COOPERNAUT

### 3.1. Problem Statement

Our goal is to learn a closed-loop policy that controls an autonomous ego vehicle, which receives LiDAR observations $O_t^{(\text{ego})}$ at time $t$. Assume that there exist a variable number of $N_t$ neighboring vehicles in the range of V2V communications at time $t$, where $O_t^{(i)}$ is the raw 3D point cloud from the onboard LiDAR of the $i$-th vehicle. The cooperative driving policy for the ego vehicle is to find a policy $\pi(a_t|O_t^{(\text{ego})}, O_t^{(1)}, \ldots, O_t^{(N_t)})$ that makes control decisions $a_t$ based on the joint observations of the ego vehicle and the $N_t$ neighboring vehicles. Here $\pi$ is parameterized by a deep neural network and trained end-to-end. In principle, we can transmit all cross-vehicle observations to the ego vehicle and process them as a whole. In practice, we have to take into account the networking bandwidth limit, which only allows for the message size orders of magnitude smaller. We thus first process the raw point clouds into compact representations, which can be transmitted through the V2V channels in real-time.

### 3.2. Background: Point Transformer

Our model's backbone is the Point Transformer [46], a newly-developed neural network structure that learns compact point-based representations from 3D point clouds. It reasons about non-local interactions among points and produces permutation-invariant representations, making itself effective in aggregating multi-vehicle point clouds. Here we provide a brief review of Point Transformers.

We adopt the same design as Zhao et al. [46], which uses *vector self-attention* to construct the Point Transformer Layer. We also apply subtraction between features and append a position encoding function $\delta$ to both the attention vector $\gamma$ and the transformed features $\alpha$:

$$y_i = \sum_{x_j \in \mathcal{X}(i)} \rho(\gamma(\phi(x_i) - \psi(x_j) + \delta)) \odot (\alpha(x_j) + \delta) \quad (1)$$

Here the $x_i$ and $x_j$ are input features of the point $i$ and $j$ respectively, $y_i$ is the output attention feature for point $i$, and $\mathcal{X}(i)$ represents the set of points in the neighborhood of $x_i$; $\phi, \psi$ and $\alpha$ are point-wise feature transformations, an MLP; $\gamma$ is an MLP mapping function with two layers and one ReLU non-linearity; $\delta$ is a position-encoding function and $\rho$ is a normalization function *softmax*. Given the 3D coordinates $p_i$, $p_j \in \mathbb{R}^3$ for point $i$ and $j$, the position-encoding function is formulated as follows:

$$\delta = \theta(p_i - p_j) \quad (2)$$

where $\theta$ is an MLP with two linear layers and one ReLU.

A *Point Transformer block* is shown in Figure 2, which integrates the self-attention layer, linear projections, and a residual connection. The input is a set of 3D points $p$ with a feature $x$ of each point. This block enables local information exchange among points, and produces new feature vectors for each point. The *down-sampling block* in Figure 2 is to reduce the cardinality of the point sets. We perform farthest point sampling [17] to the input set to obtain a well-spread subset, and then use kNN graph and (local) max pooling in the neighborhood to further condense the information to smaller sets of points. The output is a subset of the original input points with new features.

### 3.3. Our Model

We use cross-vehicle perception to augment the sensing capabilities of the ego vehicle for it to make more informed decisions under challenging situations. The key challenge is to transmit sensory information efficiently through realistic V2V channels, understand the traffic situation from the aggregated information, and determine the reactive driving action in real-time.

Our COOPERNAUT model, illustrated in Figure 2, is composed of a Point Encoder for each neighboring V2V vehicle to encode its sensory data into compact messages, a Representation Aggregator to integrate the messages from neighboring cars with the ego perception, and a Control Module which translates the integrated representations to driving commands.

**Point Encoder.** To reduce communication burdens, every V2V vehicle processes its own LiDAR data locally and encodes the raw 3D point clouds into keypoints, each associated with a compact representation learned by the Point Transformer blocks. We construct the encoder with three Point Transformer blocks accompanied by two down-sampling blocks, both with a downsampling rate of $(1, 4, 4)$. The final cardinality of intermediate representations is $P/16$, where $P$ is the number of points in the raw point cloud. In our experiments, we preprocess 65,536 raw LiDAR points to 2,048 points via voxel pooling, *i.e.*, representing the points in a voxel grid using their voxel centroid.

The message $M_j$ produced by the $j$-th vehicle comprises a set of position-based representations $M_j$ and is mathematically described as $M_j = \{(p_{jk}, R_{p_{jk}})\}^K$, where $p_{jk} \in \mathbb{R}^3$ for $k = 1, \ldots, K$ is the position of a keypoint in 3D space and $R_{p_{jk}}$ is its corresponding feature vector produced by the Point Encoder. We limit the size of $M_j$ to be at most $K$ tuples. These keypoints carrying features are in each vehi-

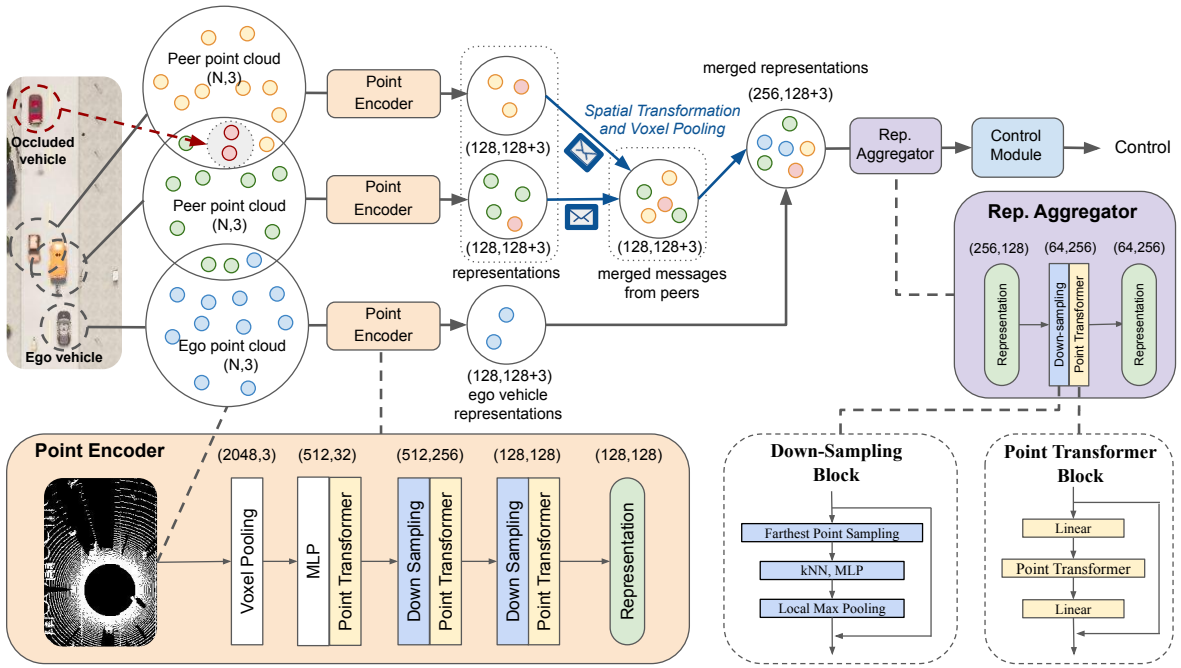


Figure 2. COOPERNAUT is an end-to-end vision-based driving model for networked vehicles. It contains a *Point Encoder* to extract critical information locally for sharing, a *Representation Aggregator* for merging multi-vehicle messages, and a *Control Module* to reason about the joint messages. Each message produced by the encoder has 128 keypoint coordinates and their associated features. The message is then spatially transformed into the ego frame. The ego vehicle merges incoming messages and compute aggregated representations through voxel max-pooling. Finally, the aggregator synthesizes joint representations from the ego vehicle and all its neighbors before passing them to the Control Module to produce control decisions. The numbers in parentheses denote data dimensions.

cle's local frame. They preserve the spatial information as their coordinates are sampled from raw point clouds.

**Representation Aggregator.** Messages transmitted from other vehicles need to be fused and interpreted by the ego vehicle. The Representation Aggregator (RA) for cooperative perception is implemented as a voxel max-pooling operation and a point transformer block. RA first spatially transforms the keypoints in other vehicles' coordinates to the ego vehicle's frame using their relative poses. This operation assumes accurate vehicle localization (*e.g.*, using HD maps). It then aggregates the incoming messages that are spatially close via max-pooling all the points located inside the same voxel grid cell. Finally, it fuses the multi-view perception information with another Point Transformer block. The two operations above preserve the permutation invariance with respect to the ordering of other vehicles and can handle a variable number of sharing vehicles. For bandwidth control, COOPERNAUT receives messages from three randomly chosen V2V vehicles in the vicinity.

**Control Module.** The control module is a fully-connected neural network designed to make control decisions based on the received messages. These control decisions include the throttle, brake, and steering, denoted as scalar $T, B, S$ respectively. These values output from the model are first clipped to their valid ranges (*e.g.*, [0,1] for throttle). To comply with the speed limit rules, we apply a PID speed controller to prevent speeding.

### 3.4. Policy Learning

We train our model to imitate the expert policy with privileged information using DAgger [35]. To warm-start policy learning, we first train the model using behavior cloning.

**Behavior Cloning.** Behavior Cloning is designed to minimize the distribution gap between the training policy and the expert policy. The goal is to find an optimal policy $\hat{\pi}$ such that the loss w.r.t. the expert's policy $\pi_{\text{expert}}$, under its induced distribution of states $S$ is minimized, *i.e.*,

$$\hat{\pi} = \arg\min_{\pi \in \Pi} \mathbb{E}_{s \sim S}[\ell_{\text{control}}(\pi(s), \pi_{\text{expert}}(s))]. \quad (3)$$

The objective function $\ell_{\text{control}}$ is a linear combination of $\ell_1$-loss of throttle, brake, and steering between the policy's actions and the expert's actions:

$$\ell_{\text{control}} = \eta_1 \ell_{\text{throttle}} + \eta_2 \ell_{\text{brake}} + \eta_3 \ell_{\text{steer}} \quad (4)$$

where $\eta_1, \eta_2, \eta_3$ are the coefficients of the loss for each action. All three coefficients are set to 1 in our experiments.

**DAgger.** Limitations of behavior cloning for autonomous driving have been discussed in Codevilla et al. [15]. DAgger [35] address the covariance shift issues via online training. The core idea is to let the student policy interact with

the environment under the supervision of the expert and record the expert's actions on the same states visited by the student. The training dataset is iteratively aggregated, using a mixture of the student's and expert's actions. The sampling policy $\pi_i$ for the $i$-th iteration follows:

$$\pi_i = \begin{cases} \pi_{\text{expert}}, & \text{w.p. } \beta_i \\ \pi_{\text{student},i}, & \text{w.p. } 1-\beta_i \end{cases} \quad (5)$$

where $\beta_i = \beta_0 \times \beta_{i-1}$ are exponentially decreasing from the initial $\beta_0$, representing the probability that the expert's action is executed at the $i$-th iteration.

### 3.5. Implementation Details

When more than three neighboring vehicles send messages, we randomly select messages from three of the vehicles. All the neighbors encode their processed point clouds locally by the 3-block Point Encoder and send the messages of size $128 \times (128, 3)$ and warp the coordinates to the ego frame. We aggregate the merged representations by another block of Point Transformer. After global max pooling, the features are concatenated with the ego speed feature before passing to the fully connected layer.

Our model has a 90ms latency on an NVIDIA GTX3090 GPU, where the point encoder takes 80ms. Our model training consists of two stages: behavior cloning and DAgger. We first train every scenario-specific model by behavior cloning, then the final policy of behavior cloning serves as an initial student policy for DAgger. We collect 4 new trajectories and append them to the Dagger dataset every 5 epochs using a sampling policy (see §3.4) with $\beta_0 = 0.8$ during the DAgger stage. For all data used for training, 25% of them are collected under accident-prone scenarios (with an occluded collider vehicle inserted) and 75% of them are normal driving trajectories. For more details, please refer to our supplementary materials and project website.

## 4. AUTOCASTSIM

We present AUTOCASTSIM, a simulation framework which offers network-augmented autonomous driving simulation on top of CARLA [16]. This simulation framework allows custom designs of various traffic scenarios for training and evaluating cooperative driving models. The simulated vehicles can be configured with realistic wireless communications. It also provides a path planning-based oracle expert with access to privileged environment information.

### 4.1. Scenarios

We designed three challenging traffic scenarios, shown in Figure 3, in AUTOCASTSIM as our evaluation benchmark. These scenarios are selected from the pre-crash typology of the US National Highway Traffic Safety Administration (NHTSA) [26], where limited line-of-sight sensing affects driving decisions:

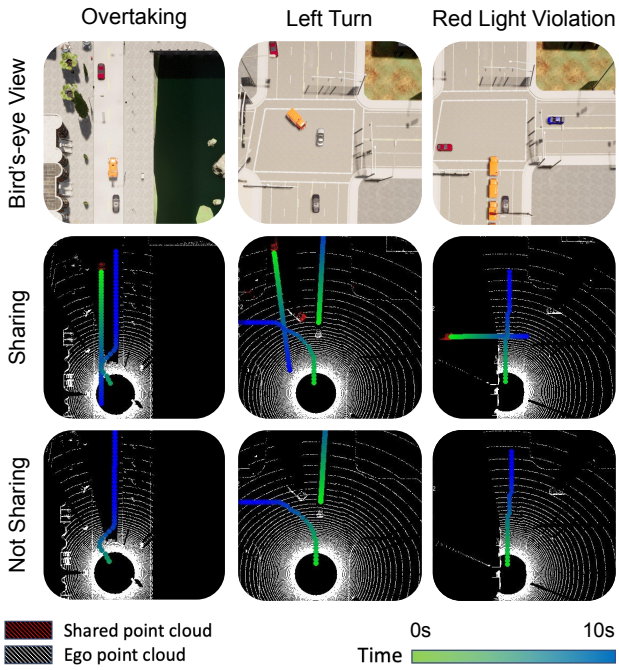


Figure 3. Benchmarking scenarios in AUTOCASTSIM. The grey car is the ego vehicle controlled by our model. The orange trucks are large vehicles that partially block views of the environment. The red car is not networked and likely to collide with the ego vehicle. All other vehicles are background traffic either with or without sharing capability. The green-blue dots mark the planned temporal trajectories for any moving vehicle, with green dots being waypoints closer in the future than the blue dots. If two planned trajectories intersect at a similar color (time), it indicates that a collision may happen. For every scenario, an RGB bird's-eye view (BEV), an ego-centric LiDAR BEV image, and a multi-vehicle fused LiDAR BEV image are presented (left to right). We use less background traffic here for illustration, and will study the effect of traffic density in §5.3.

*** Overtaking.** A truck blocks the way of a sedan in a two-way single lane road with a dashed yellow lane divider. The truck also impedes the sedan's view of the opposite lane. The ego car has to overtake with a lane change maneuver.

*** Left Turn.** The ego car tries to turn left on a left-turn yield light but encounters another truck in the opposite left-turn lane, blocking its view of the opposite lanes and potential straight-driving vehicles.

*** Red Light Violation.** The ego car is crossing the intersection when another vehicle is rushing the red light. LiDAR fails to sense the other vehicle because of the lined-up vehicles waiting for the left turn.

### 4.2. V2V Communication

To simulate realistic wireless communication, we use real V2V wireless radios to profile wireless bandwidth ca-

| | DSRC | C-V2X | 802.11n | 802.11ac |
|---|---|---|---|---|
| Throughput (Mbps) | 2.0 | 7.2 | $\sim 200$ | $\sim 900$ |
| Packet Loss (%) | $< 5$ | $< 5$ | $> 90$ | $> 90$ |
| Mobility support | Yes | Yes | No | No |

Table 1. Measured wireless throughput and packet loss rate using off-the-shelf wireless radios.

pacity and packet loss rate due to channel diversity between mobile agents. Specifically, we use three iSmartways DSRC radios and three C-V2X radios [2], mounted on top of moving vehicles, to measure the maximum capacity of continuous wireless transmission in practice. Table 1 shows the tested throughput and packet loss. It also shows the throughput of WiFi (802.11n, ac) for context. Note that the 802.11 series is not designed for mobile scenarios. Table 1 shows that V2V bandwidth is two orders of magnitude smaller than the indoor wireless capacity. The extremely limited bandwidth, in practice, poses significant challenges for designing the representations for V2V communication. We use the Winner II wireless channel model [25] in our simulator and use the measured C-V2X radio capacity and packet loss rate in the channel model. We refer to prior work [33] for the design and implementations of the coordination, scheduling, and the network transport layer.

### 4.3. Oracle Expert

The expert has access to the privileged information of the traffic scenarios. The information includes the point cloud from the LiDARs of all neighboring vehicles and the positions and speeds of these neighboring vehicles and other traffic participants. The expert transforms all of the point clouds from neighboring cars to its ego perspective (which is impractical due to the wireless bandwidth limit mentioned above). The transformed point cloud is fused for downstream obstacle detection and planning. The expert policy leverages all information above to analyze and avoid possible collisions. The path planning algorithm uses an A* trajectory planner [19] with pose and distance heuristics. The expert moves at a target speed of 20km/h.

## 5. Experiments

We first discuss the evaluation method and the experiment setup and then give a brief overview of our baselines. Next we present the main quantitative evaluation results of our methods against baselines. Finally, we provide further analysis and visualization to understand the quality of our learned model.

### 5.1. Experimental Settings

**Scenario Configuration.** We generate traces from the three scenarios we implemented in AUTOCASTSIM (§4.1) for training and evaluation. These scenarios can be programmatically re-configured with key parameters, notably the number of vehicles, vehicle spawning locations, and vehicle cruising speeds. Random combinations of these parameters are sampled to procedurally generate traces with traffic situations of varying complexity — in some cases the ego vehicle has to take emergency actions to avoid potential collisions, while in other cases, cruising along the default route can reach the destination.

**Dataset.** Specifically, for each scenario, we use the expert agent (§4.3) to generate an initial training set of 12 traces with randomized scenario configurations, followed by another randomly configured 84 traces for DAgger. In the evaluation, we systematically test each model on a spectrum of 27 randomly selected accident-prone environment configurations over three repeated runs, each using different random seeds for background traffic. For fair comparison, we use a fixed set of 27 test configurations to evaluate all models.

**Metrics.** We report three metrics, *Success Rate*, *Collision Rate*, and *Success weighted by Completion Time*:

*Success Rate (SR).* A *successful completion* of the scenario is defined as the ego agent reaching a designated target location in a permissible time without collision or prolonged stagnation. The success rate is defined as the percentage of *successful completions* among all evaluated traces.

*Collision Rate (CR).* Collision is the most common failure mode. *Collision rate* is defined as the percentage of evaluation traces where the ego vehicle collides with any entity, such as vehicles, buildings, etc.

*Success weighted by Completion Time (SCT).* SR reflects overall task success or failure. It does not differentiate the amount of time a driving agent needs to complete the traces. We introduce a third metric to weigh the success rate by the completion time ratio between the expert and the agent:

$$SCT = \mathbb{I}\{\text{agent success}\} T_{\text{expert}} / T_{\text{agent}} \tag{6}$$

where $\mathbb{I}$ is an indicator function, and $T_{\text{expert}}$ and $T_{\text{agent}}$ are the expert's and the agent's completion time, respectively. As the expert agent should require no longer completion time than the agent, the ratio resides in the range of $[0, 1]$.

### 5.2. Baselines

We compare COOPERNAUT with non-V2V and V2V driving baselines. For fair comparison, we adopt the same neighbor selection process (§3.5) among V2V approaches:

*** No V2V Sharing.** The non-sharing baseline makes decisions solely based on the onboard LiDAR data and ego speed. The model shares the same data processing scheme for an individual vehicle and point encoder architecture as our final model.

*** Early Fusion.** The Early Fusion model assumes an unrealistic communication bandwidth, with which it can trans-

| Model | Bandwidth (Mbps) | Overtaking | | | Left Turn | | | Red Light Violation | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | SR↑ | SCT↑ | CR↓ | SR↑ | SCT↑ | CR↓ | SR↑ | SCT↑ | CR↓ |
| No V2V Sharing | - | 45.3±0.6 | 43.6±0.7 | 35.8±3.6 | 40.3±5.9 | 37.8±4.6 | 55.6±9.6 | 47.3±18.7 | 46.1±18.4 | 51.4±17.4 |
| Early Fusion | 60.0 | 81.9±7.2 | 81.2±5.2 | 11.9±5.1 | 72.8±8.6 | 68.8±8.9 | 26.3±8.1 | 78.6±11.8 | 75.8±9.1 | **17.7±15.2** |
| Voxel GNN | 5.60 | 70.0±4.8 | 67.8±4.2 | 16.1±3.6 | 53.5±6.9 | 51.0±6.9 | 33.3±7.3 | 64.2±25.3 | 62.0±24.8 | 35.0±25.9 |
| COOPERNAUT (Ours) | **5.10** | **90.5±1.2** | **88.4±1.1** | **4.5±3.1** | **80.7±5.2** | **76.2±3.9** | **18.1±6.2** | **80.7±7.6** | **77.8±7.0** | **17.7±7.8** |

Table 2. Quantitative results of different models over three repeated runs. SR: Success Rate, in percentage; SCT: Success weighted by Completion Time, in percentage; CR: Collision Rate, in percentage; In the Bandwidth column, we report the communication throughput required without data compression. The bandwidth is calculated by assuming 10 Hz LiDAR scanning frequency.

mit and fuse the entire raw point cloud data from all neighboring vehicles. While this method is intractable in practice, it serves as a baseline to examine our point-based architecture's effectiveness in representation learning. To fit this model in GPU memory, we limit the size of the fused input points to 4,096. Like the previous baseline, Early Fusion also uses a 3-block Point Transformer encoder.

*** Voxel GNN.** We adapt V2VNet [41], which is designed for 3D detection and motion forecasting, to learn end-to-end driving. Every vehicle processes its local point cloud onboard and shares a voxel representation with the ego vehicle for control. It uses a graph neural network (GNN) in the ego frame as the aggregator. The control actions are predicted from the GNN-fused representations.

For fair comparison, all baselines and proposed approaches are independently trained over three repeated runs with the same training parameters (§3.5). We report the average performance over the three runs on the same scenario configurations (§5.1).

### 5.3. Quantitative Results

This section presents the empirical evaluations of all the models in the three benchmarking scenarios.

**Scenario Completion.** Table 2 shows the performance comparisons in each of the three traffic scenarios. In all three scenarios, the No V2V Sharing model has performed poorly, with less than 50% success rate for each scenario and high collision rates. All three cooperative driving models, including Early Fusion, Voxel GNN, and COOPERNAUT, have achieved substantially higher SR and SCT scores and lower collision rates than the No V2V Sharing baseline. It indicates that the V2V communication provides critical information about the traffic situation over the ego vehicle's line-of-sight sensing to make informed driving decisions. The Early Fusion method improves over the non-V2V baseline over 30% in average success rate. However, the Early Fusion baseline requires transmitting raw point clouds across vehicles, leading to an unrealistic bandwidth requirement of 60Mbps (before data compression).

In contrast, pre-processing raw sensory data into representations has dramatically reduced the bandwidth requirements while improving driving performances. Both Voxel GNN and COOPERNAUT perform sensory fusion on the representation level. In comparison to the other cooperative driving models, COOPERNAUT outperforms both Early Fusion and Voxel GNN baselines for all three scenarios. We hypothesize that the point-based representation learning of COOPERNAUT makes it robust to localization errors compared with fusing raw points in Early Fusion. Furthermore, the explicit representation of point 3D locations and the point sampling module of COOPERNAUT retain a high spatial resolution of its intermediate representations in contrast to the voxel-based feature maps used by Voxel GNN.

**Bandwidth Requirement.** As shown in Table 2, sharing raw point cloud at the LiDAR scanning rate of 10fps would require a wireless bandwidth of 60Mbps, far beyond the achievable bandwidths in the current (DSRC) and future (C-V2X or LTE-direct) V2V communication technology (expected less than 10Mbps, see Table 1). V2VNet [41] claims a bandwidth requirement of 25 Mbps with point cloud compression, which is also beyond what current V2V radios can support. In our design, both Voxel GNN and COOPERNAUT requires less than 6Mbps bandwidth, a 4× reduction of the communication data sizes of V2VNet without compression.

When developing the V2V models, we carefully explored the design space of the sharable representation size and its bandwidth requirement for both Voxel GNN and COOPERNAUT. For example, if COOPERNAUT were to share a 32×32 representation, it only needs 0.9 Mbps. However, the coarse information is insufficient for the model to attain a good performance. We find that a 128×128 point representation meets the bandwidth requirements (Table 1) without substantial performance degradation.

**Sensitivity to Traffic Density.** We further test COOPERNAUT under varied traffic densities in the most challenging Left Turn scenario. Figure 4 shows that our method generalizes to variable traffic densities, consistently outperforming the No V2V Sharing baseline. Qualitatively, we observe that No V2V Sharing drives slower in denser traffic, reacting better to emergency situations. In contrast, V2V methods do not improve much in denser traffic, as they tend to be impacted by the increased stochasticity of incoming messages from changing neighbors. Nonetheless, COOPERNAUT outperforms the baselines in all traffic densities with over 30% higher success rates than No V2V Sharing.

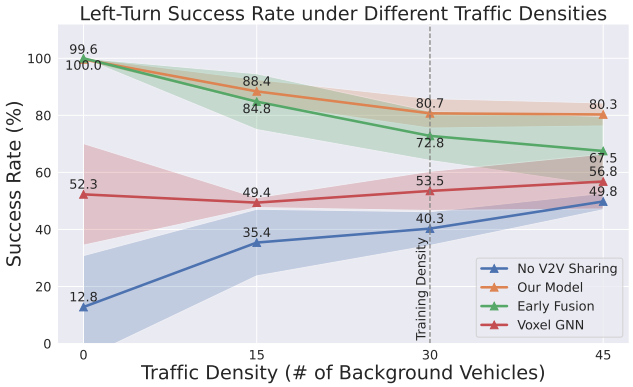


Figure 4. Sensitivity analysis on the varying levels of traffic densities in the Left Turn scenario.

**Qualitative Visualizations.** Figure 5 shows an example evaluation trajectory from Left Turn. The left-turning ego vehicle (grey) can proactively avoid collision by yielding to the opposite-going cars with COOPERNAUT. A common failure pattern of the non-sharing model is that it drives ahead to its target location regardless of any traffic violators or potential colliders due to the limited line-of-sight of its ego perception. The transmitted messages through V2V channels help our model resolve the ambiguity with cross-vehicle perception, leading to safer driving decisions in this accident-prone situation.

### 5.4. Limitations and Future Work

While our cooperative perception model conforms to realistic wireless bandwidth, we do not take into account practical networking issues, including transmission latency, networking protocols, and repetitive or lost packets. Nonetheless, COOPERNAUT is robust to packet loss to a certain extent (5% as configured in AUTOCASTSIM). Its random neighbor selection also adds another layer to endure packet loss from individual transmitters.

Furthermore, highly accurate vehicle localization is assumed, which is used by COOPERNAUT to transform the point-based representations from neighboring vehicles to the ego vehicle, even though AUTOCASTSIM simulates slight errors in the pose and height estimation of a vehicle. In reality, without a high definition map (HDMap), localization error can yield up to meter-level displacement. Using HDMap can significantly improve location and pose estimation, which is commonly adopted in both industry and academia [23, 44].

For fair comparison, we use the same down-sampling scheme for all point-based baselines and our approach, which proves to be effective in our scenarios with moving vehicles and large obstacles. For smaller objects like pedestrians, adaptive sampling schemes based on semantic information is a promising direction for future work. We would also like to extend the model architecture of COOPERNAUT to better incorporate temporal information for improving driving performance.


Figure 5. Comparison of trajectories in the Left Turn scenario. The grey car in the pictures is the controllable ego vehicle. The red car is going straight in the opposite direction, occluded behind the orange truck. Our model avoids the collision as it is able to see the red light violating vehicle from cooperative perception (highlighted in the yellow box).

## 6. Conclusion and Future Work

This work investigates vision-based driving using cooperative perception for networked vehicles in a newly designed simulation benchmark AUTOCASTSIM. We introduce COOPERNAUT, an end-to-end driving policy that encodes, aggregates, and analyzes 3D LiDAR data from networked vehicles. The point encoder and representation aggregator of COOPERNAUT retain detailed spatial information and are robust to a varying number of communicating vehicles. Our empirical results show that our method improves the robustness of autonomous driving policies in risk-sensitive traffic scenarios.

This work has ample room for future extension. Our method relies on a hand-engineered oracle for imitation learning. It leaves open questions to investigate adaptive strategies of *when* to communicate, *what* to encode in messages, and *how* to drive cooperatively, ideally without the need of an algorithmic oracle.

**Acknowledgements** This work has taken place in the Robot Perception and Learning Lab (RPL) and Learning Agents Research Group (LARG) at UT Austin. RPL research is supported in part by NSF (CNS-1955523, FRR-2145283), the MLL Research Award from the Machine Learning Laboratory at UT-Austin, and the Amazon Research Awards. LARG research is supported in part by NSF (CPS-1739964, IIS-1724157, FAIN-2019844), ONR (N00014-18-2243), ARO (W911NF-19-2-0333), DARPA, Lockheed Martin, GM, Bosch, and UT Austin's Good Systems grand challenge. Peter Stone serves as the Executive Director of Sony AI America and receives financial compensation for this work. The terms of this arrangement have been reviewed and approved by the University of Texas at Austin in accordance with its policy on objectivity in research.

## References


[1] Cloud enhanced open software defined mobile wireless testbed for city-scale deployment (cosmos). URL https://cosmos-lab.org/. 2

[2] ismartways performance measurement. URL https://fccid.io/2AQQ3IM2RSE/Test-Report/FCC-Part22-4039626. 6

[3] 18 awesome innovations in the new merceds e-class, 2016. URL https://www.businessinsider.com/mercedes-e-class-2017-features-2016-6. 2

[4] Toyota's v2v move shows industry still interested in cars talking to each other, 2018. URL https://www.consumerreports.org/automotive-technology/toyota-v2v-vehicle-to-vehicle-communications/. 2

[5] Ann Arbor Connected Vehicle Test Environment, 2019. URL http://www.aacvte.org. 2

[6] Shunsuke Aoki, Takamasa Higuchi, and Onur Altintas. Cooperative perception with deep reinforcement learning for connected vehicles. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 328–334. IEEE, 2020. 1, 2

[7] Mayank Bansal, Alex Krizhevsky, and Abhijit Ogale. Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst. *arXiv preprint arXiv:1812.03079*, 2018. 2

[8] Myra Blanco, Jon Atwood, Sheldon M Russell, Tammy Trimble, Julie A McClafferty, and Miguel A Perez. Automated vehicle crash rate comparison using naturalistic data. Technical report, Virginia Tech Transportation Institute, 2016. 1

[9] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016. 1

[10] Chenyi Chen, Ari Seff, Alain Kornhauser, and Jianxiong Xiao. Deepdriving: Learning affordance for direct perception in autonomous driving. In *Proceedings of the IEEE international conference on computer vision*, pages 2722–2730, 2015. 1

[11] Dian Chen, Brady Zhou, Vladlen Koltun, and Philipp Krähenbühl. Learning by cheating. In *Conference on Robot Learning*, pages 66–75. PMLR, 2020. 1, 2

[12] Dian Chen, Vladlen Koltun, and Philipp Krähenbühl. Learning to drive from a world on rails. In *arXiv preprint arXiv:2105.00636*, 2021. 2

[13] Qi Chen, Sihai Tang, Qing Yang, and Song Fu. Cooper: Cooperative perception for connected autonomous vehicles based on 3d point clouds. In *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, pages 514–524. IEEE, 2019. 1, 2

[14] Felipe Codevilla, Matthias Müller, Antonio López, Vladlen Koltun, and Alexey Dosovitskiy. End-to-end driving via conditional imitation learning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4693–4700. IEEE, 2018. 2

[15] Felipe Codevilla, Eder Santana, Antonio M López, and Adrien Gaidon. Exploring the limitations of behavior cloning for autonomous driving. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9329–9338, 2019. 2, 4

[16] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. *arXiv preprint arXiv:1711.03938*, 2017. 1, 5

[17] Yuval Eldar, Michael Lindenbaum, Moshe Porat, and Yehoshua Y Zeevi. The farthest point strategy for progressive image sampling. *IEEE Transactions on Image Processing*, 6(9):1305–1315, 1997. 3

[18] L. Gallo and J. Harri. Short paper: A lte-direct broadcast mechanism for periodic vehicular safety communications. In *IEEE Vehicular Networking Conference 2013*, pages 166–169. IEEE, Dec 2013. doi: 10.1109/VNC.2013.6737604. 2

[19] P. E. Hart, N. J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968. 6

[20] J. B. Kenney. Dedicated short-range communications (dsrc) standards in the united states. *Proceedings of the IEEE*, 99(7):1162–1182, July 2011. ISSN 0018-9219. doi: 10.1109/JPROC.2011.2132790. 2

[21] Seong-Woo Kim, Wei Liu, Marcelo H Ang, Emilio Frazzoli, and Daniela Rus. The impact of cooperative perception on decision making and planning of autonomous vehicles. *IEEE Intelligent Transportation Systems Magazine*, 7(3):39–50, 2015. 1, 2

[22] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars:

Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12697–12705, 2019. 2

[23] Qi Li, Yue Wang, Yilun Wang, and Hang Zhao. Hdmapnet: A local semantic map learning and evaluation framework, 2021. 8

[24] Yiming Li, Shunli Ren, Pengxiang Wu, Siheng Chen, Chen Feng, and Wenjun Zhang. Learning distilled collaboration graph for multi-agent perception. In *Thirty-fifth Conference on Neural Information Processing Systems (NeurIPS 2021)*, 2021. 2

[25] Juha Meinilä, Pekka Kyösti, Tommi Jämsä, and Lassi Hentilä. Winner ii channel models. In *Radio Technologies and Concepts for IMT-Advanced*. 2009. 6

[26] Wassim G Najm, Raja Ranganathan, Gowrishankar Srinivasan, John D Smith, Samuel Toma, Elizabeth Swanson, August Burgett, et al. Description of light-vehicle pre-crash scenarios for safety applications based on vehicle-to-vehicle communications. Technical report, United States. National Highway Traffic Safety Administration, 2013. 5

[27] D. Pomerleau. Alvinn: An autonomous land vehicle in a neural network. In *NeurPIS*, 1988. 2

[28] Aditya Prakash, Kashyap Chitta, and Andreas Geiger. Multi-modal fusion transformer for end-to-end autonomous driving. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[29] Aditya Prakash, Kashyap Chitta, and Andreas Geiger. Multi-modal fusion transformer for end-to-end autonomous driving. *arXiv preprint arXiv:2104.09224*, 2021. 1

[30] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 2

[31] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv preprint arXiv:1706.02413*, 2017. 2

[32] Hang Qiu, Fawad Ahmad, Fan Bai, Marco Gruteser, and Ramesh Govindan. Avr: Augmented vehicular reality. In *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services*, pages 81–95, 2018. 1, 2

[33] Hang Qiu, Po-Han Huang, Namo Asavisanu, Xiaochen Liu, Konstantinos Psounis, and Ramesh Govindan. Autocast: Scalable infrastructure-less cooperative perception for distributed collaborative driving. *CoRR*, abs/2112.14947, 2021. URL https://arxiv.org/abs/2112.14947. 2, 6

[34] Qualcomm. Lte direct proximity services, 2019. URL https://www.qualcomm.com/invention/technologies/lte/direct. 2

[35] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635, 2011. 4

[36] Axel Sauer, Nikolay Savinov, and Andreas Geiger. Conditional affordance learning for driving in urban environments. In *Conference on Robot Learning*, pages 237–252. PMLR, 2018. 2

[37] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10529–10538, 2020. 2

[38] R. Stahlmann, A. Festag, A. Tomatis, I. Radusch, and F. Fischer. Starting european field tests for car-2-x communication: the drive c2x framework. In *ITS World Congress and Exhibition*, Oct 2011. 2

[39] Marin Toromanoff, Emilie Wirbel, and Fabien Moutarde. End-to-end model-free reinforcement learning for urban driving using implicit affordances. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2

[40] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 2

[41] Tsun-Hsuan Wang, Sivabalan Manivasagam, Ming Liang, Yang Bin, Wenyuan Zeng, James Tu, and Raquel Urtasun. V2VNet: Vehicle-to-vehicle communication for joint perception and prediction. In *ECCV*, 2020. 2, 7

[42] Huazhe Xu, Yang Gao, Fisher Yu, and Trevor Darrell. End-to-end learning of driving models from large-scale video datasets. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2174–2182, 2017. 1

[43] Runsheng Xu, Hao Xiang, Xin Xia, Xu Han, Jinlong Liu, and Jiaqi Ma. Opv2v: An open benchmark dataset and fusion pipeline for perception with vehicle-to-vehicle communication. In *2022 IEEE International Conference on Robotics and Automation (ICRA)*, 2022. 2

[44] Bin Yang, Ming Liang, and Raquel Urtasun. Hdnet: Exploiting hd maps for 3d object detection. In *Conference on Robot Learning*, pages 146–155. PMLR, 2018. 8

[45] Xumiao Zhang, Anlan Zhang, Jiachen Sun, Xiao Zhu, Y. Ethan Guo, Feng Qian, and Z. Morley Mao. Emp: Edge-assisted multi-vehicle perception. In *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*, MobiCom '21, page 545–558, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450383424. doi: 10.1145/3447993.3483242. URL https://doi.org/10.1145/3447993.3483242. 1, 2

[46] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip H.S. Torr, and Vladlen Koltun. Point transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 16259–16268, October 2021. 2, 3

[47] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4490–4499, 2018. 2