Leveraging Cooperative Perception for Enhanced Autonomous Driving: An End-to-End Architecture Using V2V-Enhanced Point Cloud Processing

Dhruv Patel and Wencen Wu*
Computer Engineering Department
San José State University
San José, CA, USA
Emails: dhruvsarju.patel@sjsu.edu, wencen.wu@sjsu.edu

***Abstract*—Optical sensors and learning algorithms designed for autonomous vehicles have made substantial progress in recent years. Nevertheless, the present dependability of autonomous systems is restricted due to their limited ability to sense objects directly in their line of sight and the susceptibility of data-driven techniques to malfunction in extreme circumstances. The development of modern telecommunication technology has generated interest in cooperative perception through vehicle-to-vehicle communication, specifically for improving autonomous driving in dangerous or urgent situations. This study presents a new driving architecture that covers the entire process, using a sophisticated learning model designed for cooperative driving based on visual information, utilizing inter-vehicle awareness. Our methodology efficiently compresses LiDAR data into concise point-based representations, facilitating their transfer between vehicles via realistic wireless communication channels. In order to assess the effectiveness of our method, we have created a network-enhanced framework on top of CARLA, a driving simulation platform that has been enhanced with network capabilities. This framework includes a wide range of accident-prone scenarios to thoroughly assess the performance of our model. The results of our study demonstrate a significant 40**

***Index Terms*—LiDAR, Cooperative Perception, autonomous driving**

I. Introduction

The integration of autonomous driving and advanced driver-assistance systems has been hampered by significant safety concerns. Although deep learning techniques have enhanced the autonomy stacks through data-driven approaches [9, 10, 42], policies derived from these methods exhibit fragility, particularly when confronted with rare and extreme scenarios encountered sporadically, approximately a few instances per million miles driven [8]. This vulnerability is further magnified by the inherent limitations of optical sensing technologies, such as stereo cameras and LiDAR, utilized in individual vehicles. These sensors primarily rely on line-of-sight and often perform suboptimally in adverse weather conditions. However, with the emergence of advanced telecommunication technologies like 5G and vehicle-to-vehicle (V2V) communications, the concept of cooperative perception [6, 13, 21, 32] has evolved as a promising framework. This approach facilitates the real-time sharing of sensor data among vehicles and roadside devices [45], potentially expanding each vehicle's perceptual range and enhancing the understanding of the intents and trajectories of nearby vehicles, thereby improving safety in scenarios prone to accidents.

The research work is supported by NSF grant RINGS-2148353.

Leveraging cooperative perception for autonomous driving policies should ideally build upon current deep learning paradigms that are tailored for individual vehicle perception [11, 16, 29]. This would involve treating the collective sensory data from multiple vehicles as an enhanced version of onboard sensing. The efficacy of cooperative perception depends on the efficient dissemination of data within the constraints of network capacity and the successful utilization of this aggregated data to build a comprehensive and precise comprehension of the traffic conditions. Recent studies on cooperative driving have highlighted the advantages of utilizing cross-vehicle observation to improve sensory capabilities and gather information for making driving decisions [6, 21]. However, these approaches often compress the raw sensory input into metadata that has a reduced number of dimensions. Previous studies have devised methods for merging data from 3D sensors (AVR [32], Cooper [13]) and integrating representations (V2VNet [41]) to collect information from surrounding vehicles via V2V communications. Nevertheless, the primary emphasis of these endeavors has been on scrutinizing unchanging datasets for the sake of 3D detection and motion prediction, rather than on the development of interactive driving tactics..

In this context, we present our model, a novel end-to-end cooperative driving model tailored for networked vehicles. Our model is designed to integrate encoded LiDAR data shared via V2V channels, taking into account realistic constraints on channel capacity. To ensure the transmission of crucial scene data from neighboring vehicles while adhering to bandwidth limitations, our driving policy architecture incorporates the Point Transformer [46], a self-attention mechanism tailored for point cloud processing. This design preprocesses raw point clouds locally on each vehicle into spatially-aware neural representations. These compact representations are not only bandwidth-efficient but also physically meaningful, enabling spatial transformation and integration with ego vehicle data. The fully differentiable nature of our architecture allows for the backward flow of control supervision, which imitates an expert planner with access to privileged information, thereby ensuring that the representations and transmitted messages are laden with relevant task information.

To evaluate the effectiveness of our model, we used a pre-built CARLA based stimulation environment featuring three designed scenarios that pose significant challenges for ego vehicle perception in understanding complex traffic situations. We benchmark our model against baseline no sharing models and early fusion models.

In summary, our key contributions are:

- The introduction of our model, an end-to-end cooperative driving model that utilizes realistic vehicle-to-vechile communication channels to learn compact and easily interpretable representations. These representations significantly aid the ego vehicle in enhancing its decision-making capabilities.

II. Related Work

Imitation Learning in Cooperative Perception.The process of training a driving controller entails the creation of closed-loop policies using deep learning methods, mostly utilizing either imitation learning or reinforcement learning.The utilization of imitation learning for autonomous driving was first established many years ago and has gradually advanced to address increasingly intricate and urban driving scenarios. In recent times, reinforcement learning has been a prominent method, showcasing its potential to create better strategies for complex driving scenarios. Significantly, reinforcement learning typically necessitates a large amount of data and the creation of a complex reward mechanism. Our approach utilizes imitation learning, where we rely on an expert oracle with extensive worldwide knowledge to improve the effectiveness of training.

Advancements in 3D Perception for Autonomous Vehicles. The utilization of 3D perception in autonomous driving has become more popular, thanks to the decreased expenses of LiDAR sensors. Initial efforts in 3D object detection in this domain have resulted in the creation of more complex models and advanced methodologies. One of the recent advancements is the investigation of end-to-end driving utilizing data obtained from point clouds. There are two main approaches in the field of 3D perception architectures: one way turns point data into voxels, while the other method directly analyzes point coordinates. Our model utilizes a transformer-based structure that emphasizes point-based techniques, ensuring a high level of spatial detail and reducing the amount of data transfer required. As a result, there is no need for compression, which was necessary in earlier implementations.

The Role of Cooperative Perception in Autonomous Vehicles The integration of network connectivity into vehicles presents substantial benefits for enhancing the safety and efficacy of autonomous driving. Modern vehicles are increasingly equipped with systems enabling the exchange of environmental data through vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2X) communications. These communications are enabled by technologies such as Dedicated Short Range Communication (DSRC) and cellular-assisted V2X, which are now standard features in new vehicle models. City-scale academic and research platforms have been established to assess the feasibility of these networked and cooperative vehicle systems. Prior research has introduced cooperative perception systems that extend a vehicle's perceptual range by sharing visual information among nearby vehicles by optimizing for scalability in congested traffic areas by using edge computing, road side infrastructures and ad-hoc networking methods. In contrast, our focus lies in making using of vehicles and their onboard sensors to communicate and cooperate and navigate effectively in real world envirnoments.

III. Problem Formulation

Our objective is to develop a model that governs an autonomous ego vehicle, receiving LiDAR data $D_t^{(\mathrm{ego})}$ at each time point t. Suppose there are V_t neighboring vehicles within the V2V communication range at any given time t, with $D^{(i)}t$ representing the raw 3D point cloud captured by the LiDAR of the i-th vehicle. The task of the cooperative driving model for the ego vehicle involves deriving a function $\pi(a_t | Dt^{(\mathrm{ego})}, D_t^{(1)}, \ldots, D_t^{(V_t)})$ that bases driving actions a_t on the collective observations from both the ego and the V_t neighboring vehicles. In this context, the policy, represented by π, is constructed using a deep neural network and is trained entirely in an end-to-end fashion. While the ideal scenario would involve directly transferring all cross-vehicle observations to the ego vehicle for processing, practical constraints such as network bandwidth limitations require that these observations are initially transformed into condensed representations.

IV. Background

Point Transformer Our model integrates the Point Transformer, a contemporary neural network architecture designed to efficiently extract compressed representations from 3D point clouds. This architecture adeptly handles non-local interactions among points and delivers permutation-invariant outputs, a crucial feature for seamlessly integrating point clouds from diverse sources, such as multiple vehicles. Below is a brief summary of the Point Transformer's capabilities.

Our implementation employs *vector self-attention* within the Point Transformer Layer. We applied feature subtraction and enhance both the attention vector γ and the transformed features α with a position encoding function δ:

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

In this formula, x_i and x_j denote the input features for points i and j respectively, with y_i being the resultant attention feature for point i. $\mathcal{X}(i)$ indicates the set of neighboring points around x_i. The transformations ϕ, ψ, and α are implemented through point-wise feature transformations using an MLP; γ serves as a two-layer MLP function with a ReLU activation; δ represents a position-encoding function, and ρ is a normalization function, specifically *softmax*. The positional relationship between 3D coordinates $p_i, p_j \in \mathbb{R}^3$ of points i and j is encoded as follows:

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

Here, θ is an MLP that includes two linear layers and a ReLU activation.

Voxel-based and Point-based object detection methods represent two distinct approaches to handling 3D data for object detection tasks. Voxel-based methods, such as VoxelNet, SECOND, and PointPillars, transform point cloud data into a structured voxel grid. This voxelization process allows the use of 3D convolutions, making it possible to leverage powerful convolutional neural networks (CNNs) for feature extraction and object detection. The regular grid structure of voxels simplifies the computational process and enables efficient handling of large-scale environments, crucial for applications like autonomous driving and robotics. However, voxel-based methods can suffer from issues related to resolution and data sparsity, as high-resolution voxel grids are computationally expensive and sparse data can lead to inefficiencies.

On the other hand, **point-based methods**, such as CornerNet, ExtremeNet, and CenterNet, focus on detecting key points or landmarks directly from the point cloud data without transforming it into a grid structure. These methods often offer higher precision by identifying specific parts or features of objects, making them particularly useful for detailed tasks like human pose estimation or tracking objects with irregular shapes. Point-based methods can more accurately capture fine details and object structures, but they can be computationally intensive due to the need for detecting multiple key points and managing the irregular nature of point cloud data. While voxel-based methods excel in efficiency and scalability, point-based methods provide superior precision and flexibility in representing complex object shapes.

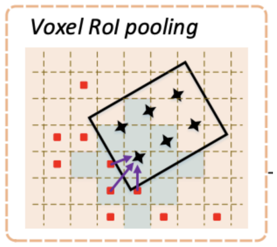


Figure 1: Voxel RoI Pooling

V. Model

In our methodology, we utilize cross-vehicle perception to enhance the sensing capabilities of the ego vehicle by augmenting it with sensed data of neighbouring vehicles, enabling it to make more knowledgeable and informed decisions, particularly in obstructed and unsignaled intersections driving situations. The main obstacle lies in effectively transmitting sensory data through practical Vehicle-to-Vehicle (V2V) communication channels taking into account the communication loss in trasnsit and other external factors, interpreting the collective traffic conditions derived from this combined data, and promptly determining appropriate driving actions in real-time. Our model, as depicted in Figure 2, addresses this challenge by integrating various components: LiDAR Point Encoder for Neighboring V2V Vehicles: Each neighboring V2V vehicle is equipped with its LiDAR Point Encoders, which are responsible for compressing its sensory data into concise permutation messages with information loss. These messages encapsulate crucial information regarding the vehicle's perception of its surroundings.

Representation Aggregator: This component merges the messages from neighboring V2V vehicles with the ego vehicle's own perception. By combining information from multiple sources, the ego vehicle gains a more comprehensive understanding of its environment.

Control Module: The integrated data from the Representation Aggregator is fed into the Control Module, which processes this information and generates appropriate driving commands. These commands dictate the actions the ego vehicle should take based on the collective perception of the environment. In order to streamline communication and ensure the efficient processing of sensory data, each Vehicle-to-Vehicle (V2V) vehicle independently handles its own LiDAR data locally. This process entails encoding the raw 3D point clouds into permutation invariant representations, each associated with a condensed representation acquired through Point Transformer blocks. Point encoder architecture is made up 3 Point Transformer blocks follwed by 2 downsampling blocks with rate of $(1, 4, 4)$. As a result, intermediate representations are obtained with a reduced cardinality of N/16, where N is number of points in orginal LiDAR point cloud denotes the number of points in the original point cloud. In our experimental setup, voxel pooling is employed to reduce the raw LiDAR points from 65,536 to 2,048 points. This reduction process represents the points within a voxel grid by their respective voxel centroids. By minimizing data size, we facilitate efficient transmission and processing of sensory information, while still preserving crucial details essential for accurate perception and decision-making. In summary, our model architecture and processing approach enable the effective utilization of cross-vehicle perception, thereby enhancing the ego vehicle's situational awareness and driving capabilities.

The message ME_j produced by the j-th vehicle comprises a set of position-based representations ME_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 ME_j to be at most K tuples. These keypoints carrying features are in each vehicle's local frame and preserve the local spatial information.

The ego vehicle uses cooperative perception aggregation methods, using a point transformer block and voxel max-pooling. It uses HD maps to translate keypoints from other cars' coordinates to the vehicle's reference frame. The vehicle then merges incoming messages and processes the combined multi-view perception data, ensuring permutation invariance. The control unit is networked and produces control commands like throttle, brake, and steering in the range [0, 1] with range validity controlled by PID Controller.

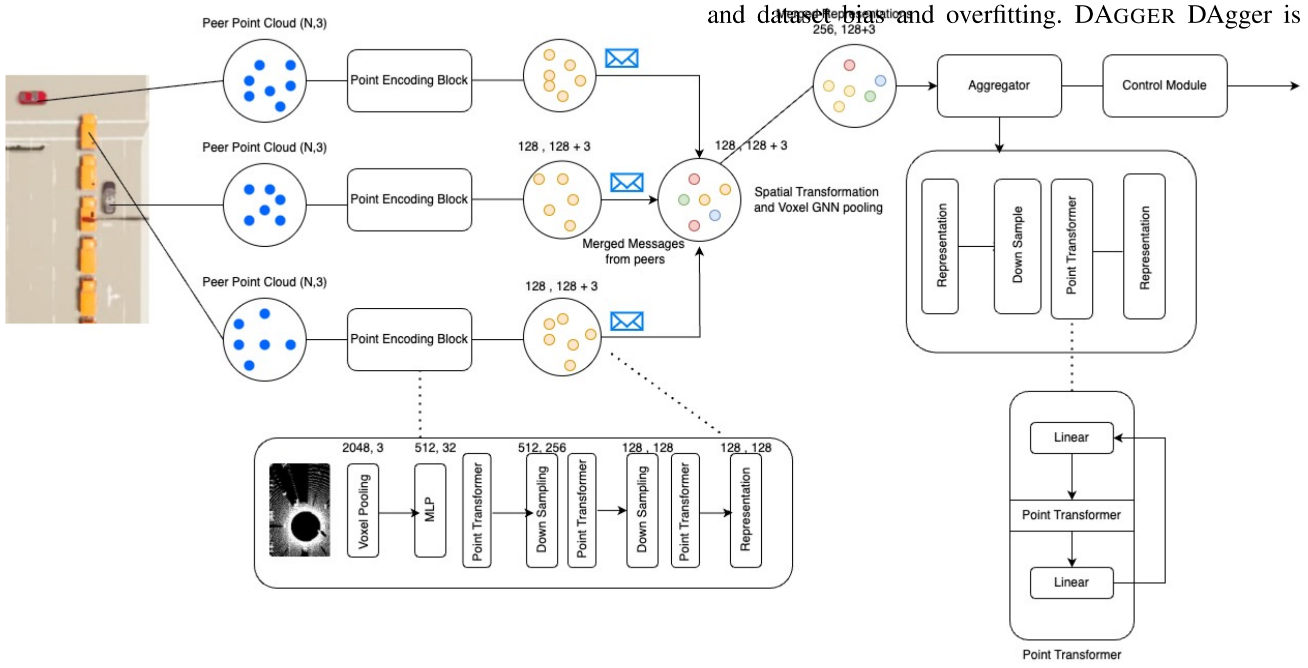


Figure 2: Architecture Of Cooperative Perception Model. Data dimensions are indicated in parentheses

VI. Imitation Learning

Reinforcement learning (RL) is a fascinating machine learning field where an agent interacts with an environment following a policy. The agent takes actions based on this policy, receives rewards, and transitions to new states, aiming to maximize long-term cumulative rewards.

While RL algorithms generally perform well, they struggle in environments with sparse rewards, such as games that only reward upon winning or losing. To mitigate this, we can manually design reward functions for more frequent feedback. However, this can be highly complex, especially in scenarios like self-driving vehicles where no direct reward function exists.

Imitation learning (IL) offers a practical solution. Instead of relying on sparse rewards or complex manual reward functions, IL uses expert demonstrations for the agent to imitate, allowing it to learn the optimal policy more effectively. Behaviour cloning (BC) is the most basic form of imitation learning. It involves employing supervised learning to learn the policy of an expert.The process of behavioural cloning is rather straightforward.

Our goal is to find a policy $\hat{\pi}$ (trained policy) mimicking π (expert policy)which minimizes the loss function defined below under its induced distribution of states, i.e.:

$$\hat{\pi} =_{\pi \in \Pi} \mathbb{E}_{s \sim d_\pi}[\ell(s, \pi)] \tag{3}$$

The objective function ℓ is a linear combination of [control , brake , throttle] loss objective functions.

Many drawbacks of the behavior cloning method render it inappropriate for simulating dynamic and complex events, such negotiating complex traffic situations. These constraints are training instabilities, additional generalization issues such handling dynamic objects and the lack of causal modeling, and dataset bias and overfitting. DAGGER DAgger is a methodical training approach that involves gathering on-policy trajectories during each iteration based on the current policy and then refining the subsequent policy based on the collective experience from all collected trajectories. The policy utilized for trajectory sampling in each iteration can be expressed as a combination of the expert policy $\hat{\pi}_e$ and the learned policy $\hat{\pi}_l$, as $\pi_i = \beta_i \pi^* + (1 - \beta_i)\hat{\pi}_l$ where is a parameter that gradually decreases across iterations.

Initially, is set to 1.

By adopting this strategy, DAgger dynamically expands the dataset by incorporating input states that the learned policy is likely to encounter during execution, based on past experiences. This strategy addresses the issue of compounding errors in iterative training by providing guidance from the expert for states where the agent diverges from optimal behavior. Below describes the Dagger Algorithm

```
Initialize $\mathcal{D} \leftarrow \emptyset$.
Initialize $\hat{\pi}_1$ to any policy in $\Pi$.
**for** $i = 1$ **to** $N$ **do**
    Let $\pi_i = \beta_i \pi^* + (1 - \beta_i)\hat{\pi}_i$.
    Sample $T$-step trajectories using $\pi_i$.
    Get dataset $\mathcal{D}_i = \{(s, \pi^*(s))\}$ of visited states by $\pi_i$
    and actions given by expert.
    Aggregate datasets: $\mathcal{D} \leftarrow \mathcal{D} \bigcup \mathcal{D}_i$.
    Train classifier $\hat{\pi}_{i+1}$ on $\mathcal{D}$.
**end for**
**Return** best $\hat{\pi}_i$ on validation.
```

Figure 3: Dagger Algorithm

VII. Model Implementation

Our method involves using a random selection process to choose messages from three surrounding vehicles when there are more than three vehicles transmitting messages. Each adjacent car independently analyzes its own point clouds using a 3-block Point Encoder and sends messages of size 128 Ã— (128, 3), while modifying the coordinates to the reference frame of the ego vehicle. Afterwards, we combine these encoded representations with an extra Point Transformer block. After performing global max pooling, the resulting features are combined with the ego speed feature and then inputted into a fully connected layer.

Our model's training comprises two independent stages: behavior cloning and DAgger. At first, we train individual models for each scenario using behavior cloning. The policy acquired by behavior cloning serves as the initial student policy for the future DAgger phase. During the DAgger phase, we collect four more trajectories and enhance the DAgger dataset every five epochs. This augmentation is accomplished through the utilization of the sampling dagger algorithm, as outlined in Figure 3, with an initial mixing value β_0 set to 0.8. This technique guarantees the ongoing enhancement of the model's performance by utilizing the combined knowledge acquired from all training data.

In order to assess the efficacy of our model, we carried out experiments in two distinct scenarios, offering comprehensive insights into its performance and suitability in practical situations. Through adhering to this methodical training methodology and thorough evaluation process, our objective is to showcase the effectiveness and resilience of our suggested technique, therefore making a valuable contribution to the progress of research in autonomous driving systems. We evaluated our model in two specific scenarios:

Overtaking another vechile in a single lane Scenario: In this scenario, there is a situation where a truck is obstructing the path of a sedan on a two-way single-lane road. The road has a dashed yellow lane divider, and the truck is blocking the view of the opposite lane for the sedan. As a result, the ego car (the car under consideration) needs to execute a lane change maneuver to overtake the truck safely. The objective here is to simulate a scenario where the ego car needs to make a lane change to pass an obstruction safely. This could happen in real-world driving situations where slower-moving vehicles obstruct the flow of traffic.

Red Light Violation by another vechile at a intersection Scenario: In this scenario, the ego car is traversing an intersection when another vehicle violates a red light. The LiDAR system, which is used for detecting obstacles and other vehicles, fails to detect the oncoming vehicle due to a line of vehicles waiting for a left turn, obstructing the view. This scenario aims to replicate a situation where the ego car encounters a safety hazard caused by another vehicle's violation of traffic rules. It highlights the challenges faced by autonomous vehicles in detecting and responding to unexpected events on the road.

Realistic Wireless Communication Simulation: To simulate realistic wireless communication between vehicles (V2V communication), actual V2V wireless radios are used to measure bandwidth capacity and packet loss rates caused by channel diversity among mobile agents. The objective of this simulation is to assess the performance of V2V communication systems in real-world conditions. By using actual V2V radios and measuring bandwidth capacity and packet loss rates, researchers can evaluate the effectiveness and reliability of V2V communication protocols. Traces from three scenarios implemented in the CARLA simulator are generated for training and evaluation purposes. These scenarios are customizable by changing various parameters like number of vehicles, locations etc. By sampling random combinations of these parameters, traces with varying traffic complexities are created. Some scenarios require the ego vehicle to execute emergency maneuvers to avoid collisions, while others involve following the default route to reach the destination. Objective: The objective here is to generate diverse sets of scenarios with varying levels of traffic complexity for training and evaluating autonomous driving algorithms. By simulating different driving situations, researchers can assess the robustness and adaptability of autonomous vehicle systems in various environments.

we present three key metrics to assess the performance of our model and compare it with baseline scenarios: The success rate is the proportion of successful completions in which the ego agent arrives at the intended place in a reasonable amount of time without running into collisions or spending too much time stationary. A successful completion suggests that the ego car meets predetermined standards for safe and effective travel by navigating to its destination.

The percentage of assessment traces in which the ego vehicle collides with any entity—other cars, people, or static objects like barricades or buildings—is known as the collision rate. Better safety performance is indicated by a reduced collision rate, which shows the model can navigate without creating mishaps or disturbances.

The Success weighted by Completion Time statistic considers the time taken to complete each scenario and adjusts the success rate accordingly. The task involves comparing the ratio of completion times between an expert and an agent. The success rate is adjusted based on the relative efficiency of reaching the target place. This statistic offers a fair assessment of both the achievement of the task goal and the efficiency in doing so.

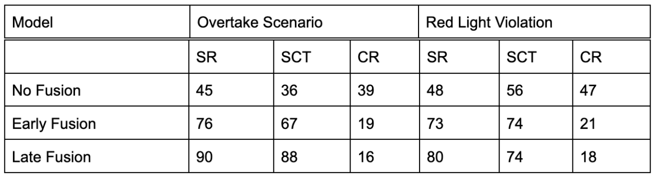
$$SCT = \mathbb{I}\{\text{agent success}\} T_{\text{expert}} / T_{\text{agent}} \tag{4}$$

To establish a baseline for comparison, we introduce two baseline scenarios: No Sharing: In this baseline model, decisions are made solely based on onboard LiDAR data and the ego vehicle's speed. This approach does not utilize information from neighboring vehicles and relies solely on local sensor data for decision-making.

Early Fusing of RAW LiDAR data: This baseline assumes an unrealistically high communication bandwidth, where all vehicles are able to transmit their entire raw point cloud and fuse received point cloud from neighboring vehicles. This approach aims to maximize information sharing but may not be practical due to bandwidth limitations in real-world scenarios.

Late Fusion of RAW LiDAR data: This baseline, initially created for 3D detection and motion forecasting, has been modified to be suitable for end-to-end driving. Every car independently analyzes its own local point cloud data on its own computational system and then transmits a voxel representation of this data to the ego vehicle for control purposes. This strategy integrates information from adjacent cars in a more organized manner in contrast to previous fusion techniques.

By comparing our model's performance with these baseline scenarios, we aim to assess its effectiveness in leveraging communication and fusion of sensor data for autonomous driving tasks. Through detailed analysis of these key metrics, we provide insights into the relative strengths and weaknesses of our proposed approach compared to baseline methods. We report three key metrics in Figure 3:



Model	Overtake Scenario			Red Light Violation		
	SR	SCT	CR	SR	SCT	CR
No Fusion	45	36	39	48	56	47
Early Fusion	76	67	19	73	74	21
Late Fusion	90	88	16	80	74	18

Figure 4: Results Comparison for models

References