Point-Cloud-based 3D Object Detection for Autonomous Navigation in Unmanned Ground Vehicles

by

SAMI SADAT
20301095
SHAOWNAK MD. IBNE SHAHRIAR TALUKDER
20101504
SHAWMIKA PROTICHI SATTAR LOGNO
18201113

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

Department of Computer Science and Engineering
Brac University
November 2024

Declaration

It is hereby declared that

1. The thesis submitted is my/our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

SAMI SADAT
20301095

SHAOWNAK MD. IBNE SHAHRIAR TALUKDER
20101504

SHAWMIKA PROTICHI SATTAR LOGNO
18201113

Approval

The thesis titled "Point-Cloud-based 3D Object Detection for Autonomous Navigation in Unmanned Ground Vehicles" submitted by

1. SAMI SADAT (20301095)
2. SHAOWNAK MD. IBNE SHAHRIAR TALUKDER (20101504)
3. SHAWMIKA PROTICHI SATTAR LOGNO (18201113)

Of Summer, 2024 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on October, 2024.

Examining Committee:

Supervisor:
(Member)

Dr. Md. Golam Rabiul Alam
Professor
Department of Computer Science and Engineering
Brac University

Program Coordinator:
(Member)

Dr. Md. Golam Rabiul Alam
Professor
Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)

Dr. Sadia Hamid Kazi
Chairperson and Associate Professor
Department of Computer Science and Engineering
Brac University

Point-Cloud-based 3D Object Detection for Autonomous Navigation in Unmanned Ground Vehicles

Abstract

Autonomous navigation for UGVs faces significant challenges in detecting objects accurately in complex environments. Despite advancements in 2D object detection, the absence of robust 3D object detection models leave a critical gap in the accurate identification of objects in real-time UGV applications. In this thesis, we propose a novel approach for 3D object detection in the context of autonomous navigation for Unmanned Ground Vehicles (UGVs). The suggested approach uses a two-stage pipeline. Utilizing the additional depth information from the 3D remote Sensor, 3D proposals are generated from the point cloud data in the initial stage. These proposals act as potential foci for the detection of objects. GLENetVR and SE SSD fusion architecture is used in the second stage to train and detect objects inside the suggested bounding boundaries. The two 3D Networks make it possible to more accurately distinguish between objects and the backdrop because they capture the spatial relationships in the volumetric representations of the point clouds. Combining two Neural Network and CNN models requires combining their feature representations, such as concatenation or element-wise combination, to form a combined feature representation used for object recognition. Through comprehensive testing and evaluation of benchmark datasets, we want to show the effectiveness and efficiency of our suggested strategy in comparison to existing 2D object detection methodologies, which are limited by their reliance on only visual information. Our research lays the door for increased safety and dependability in autonomous navigation systems for UGVs by embracing the promise of cloud point-based 3D object identification. Our proposed model has shown superior performance, achieving high accuracy of surpassing both the SE SSD and GLENetVR models.

Keywords: 3D object detection; autonomous navigation; Unmanned Ground Vehicles (UGVs); point cloud data; Convolutional Neural Networks (CNNs); 3D proposals; 3D CNN; CNN architecture fusion; Dual CNN.

Acknowledgement

Firstly, all praise to the Great Allah for whom our thesis has been completed without any major interruption.
Secondly, to our supervisor Dr. Md. Golam Rabiul Alam sir for his kind support and advice in our work. He helped us whenever we needed help.
And finally to our parents without their support, it may not be possible. With their kind support and prayer, we are now on the verge of our graduation.

Table of Contents

List of Figures

List of Tables

Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

CNN Convolutional Neural Network

$GLENet - VR$ Global and Local Enhanced Network for Visual Recognition

IoU Intersection over Union

mAP Mean Average Precision

MLP Multi-Layer Perceptron

$MTHead$ Multi-Task Head

$ReLU$ Rectified Linear Unit

SSD Single Shot MultiBox Detector

Chapter 1

Introduction

For autonomous driving, the ability to perceive and understand the three-dimensional environment around the vehicle is a benchmark. Object detection, the primary function in this industry, is central to safety, efficiency, and navigation. It allows autonomous vehicles to detect and react to various objects, from pedestrians and cars to road signs and obstacles. With lives and property at stake, the accuracy and reliability of tracking are second to none, making it a key component in autonomous vehicle tracking systems.

Traditional object recognition, usually limited to two-dimensional (2D) visuals, initially improved perceptual capacity. Real-world complexity, however, requires a more nuanced understanding. This is where 3D object recognition appears as a transformation model. Unlike its 2D counterpart, 3D object detection uses extreme depth of field, which is invaluable for accurate placement, alignment, and orientation of objects in three-dimensional space Drawing on technologies like LiDAR (Light Detection and Ranging) and Point Cloud Data to allow the UAVs to navigate autonomously. 3D object recognition models use volumetric geometry data and point cloud information [18] [15]. These models exploit the geometric properties of point clouds for accurate object detection in real-world applications [12]. Methods such as voxel-wise feature learning and 3D voxel convolution neural networks are used to generate proposals and regress offsets of 3D bounding boxes. Moreover, these models incorporate geometric information and semantic properties of point clouds to provide feature extraction and search accuracy of object offerings to improve. For flexibility, voxel-based and point-based images are combined. Some models combine 2D camera images with 3D LIDAR point clouds for better visualization. Overall, 3D object recognition models use volume geometry data and point cloud information to achieve accurate and robust object recognition.

To achieve this accurate 3D object detection, we will use the GLEnet-VR and SE-SSD models. GLEnet-VR, a pioneering architecture, leverages the power of neural networks to capture complex features from point cloud data. At the same time, the SE-SSD model uses Single Shot MultiBox Detector (SSD) technology, which is known for its real-time 2D sensing capabilities. By blending the capabilities of a 3D view of GLEnet-VR with SE-SSD functionality, we are paving the way for a complete solution. This fusion enables our system to convert from LiDAR-based point cloud data to webcam imagery in real-time, providing unmatched object detection

accuracy for autonomous vehicles in different environments.

1.1 Research Problem

In autonomous systems, accuracy and the pursuit of adaptability are of the utmost importance. Traditional 2D object recognition, a remarkable advance in quality, suffers from a fundamental limitation—it works in two dimensions, so it can't cover the fine fabric of the three-dimensional world [6]. It's not enough to render the complex three-dimensional space challenges.

Furthermore, most of the existing research mainly focuses on 2D object recognition systems, which unfortunately omits an important feature which is depth object information [4]. These systems inherently waste their effectiveness, hindering their usefulness in important areas such as intelligent video surveillance, robotic road navigation, and the growing field of autonomous driving technology [4]. As a result, The broad potential of autonomous systems remains unrealized, constrained by the inherent limitations of traditional detection methods. To meet this challenging challenge, researchers embarked on a journey into 3D object recognition by leveraging deep learning techniques [2]. This visionary approach seeks to use depth information to balance out the complexity of real-world detection. Addressing this extra dimension promises a paradigm shift of the reliability in the dynamic environments [1].

Although 3D visualization methods offer a tempting solution to the limitations of 2D visualization, it has an increased computational expense. Applying the 3D model to real-time applications such as webcam-based recognition or 2D camera-based recognition presents a formidable challenge. These models, while adept at detecting depth in controlled environments, tend to produce highly inaccurate results in the face of complex real-world challenges. Lighting conditions, image noise, and the unpredictability of object motion are major obstacles, leading to a significant drop in detection accuracy. Research shows an end-to-end, efficient pseudo-stereo 3D detection framework that uses a Single-View Diffusion Model (SVDM) to train the detector and reduce the accuracy gap between LiDAR-based and camera-based methods [19].

The solution to this challenge is the integration of complementary models. We will be using two CNN models, GLEnet-VR and SE-SSD. The deep-rooted GLEnet-VR is designed to extract complex features from the rich point cloud data. On the other hand, SE-SSD boasts the real-time strength of Single Shot MultiBox Detector (SSD) technology, which has found its stripes in 2D object recognition. Combining the strength of GLENet-VR's 3D perceptions with SE- SSD's efficiency we are a way to find a holistic solution. This hybrid model stands as a key component in object detection, evolving seamlessly from LiDAR-generated point cloud data to real-time webcam imagery. This thesis capitalizes on GLENet-VR's precise depth perception and SE-SSD's speed and adaptability. This dynamic collaboration addresses the challenges posed by real-world data and provides unparalleled object detection accuracy, even in dynamic environments and diverse scenarios.

The transition from traditional 2D to 3D emergent object recognition represents a critical moment in the development of independent systems. In our research, we will journey through two-dimensional barriers of deep understanding, which is required to navigate real-world complexity. The fusion of models in our research will make it possible to detect objects in the dynamic landscapes of our modern world for the autonomy of the UAVs.

1.2 Research Contribution

The primary objective of this research is to formulate an innovative 3D detection approach by utilizing GLE-NetVR model and SSD model to allow for the accurate object recognition for UGVs. Our research efforts lead the pioneering field of 3D object recognition, where a combination of advanced techniques and models is poised to change the landscape of autonomous systems, especially in the field of unmanned ground vehicles (UGV). At its core, our research stems from a quest for accuracy, adaptability, and real-world applicability. Our goal is to create innovative solutions that seamlessly integrate deep information into the search process, enabling autonomous systems to navigate through the complex dynamic environment with increased situational awareness and security. The research objectives include:

1. We developed a state-of-the-art 3D object recognition techniques based on the latest advances in deep learning. These methods will be the cornerstone of accurate and reliable identification.

2. We fused a model with high accuracy which has les computational complexity compared to state-of-the art architectures.

3. We contributed in new systems to seamlessly add in-depth information to the search process. This integration is necessary for a deeper understanding of object-place relationships.

4. Our model lead to a solution that facilitates the real-time visualization of 3D objects, meeting the time-critical requirements of applications such as autonomous vehicle technology.

5. Our proposed methodology goes beyond mere object recognition and enable autonomic systems to sense the spatial context of objects.

6. GLE-SSD-VR transcends domain boundaries by taking advantage of a wide range of applications from intelligent video surveillance and robotic navigation to autonomous driving.

7. We explored the fusion of two formidable models, GLEnet-VR and SE-SSD, that combine their capabilities to achieve a harmonious blend of depth perception and real-time efficiency.

8. We object recognition capabilities with our model in developed 3D offerings providing higher recognition accuracy necessary for ensuring safety in autonomous systems.

1.3 Thesis Organization

This thesis is organized into six chapters, each addressing specific aspects of the research:
In Chapter 1, we provide an overview of the research problem, and research contribution of the study. It outlines the significance of 3D object detection for autonomous navigation and highlights the challenges that current methods face. The proposed approach is briefly introduced, along with the contributions made by this research.

In chapter 2, a detailed review of existing methods and advancements in 2D and 3D object detection is presented. The chapter discusses the background of the Point cloud based 3D object detection. Also it discusses the related works and recent developments in 3D neural networks, particularly focusing on models such as SE SSD, GLENetVR, and other related architectures. This sets the stage for understanding the research gap addressed by the proposed fusion model.

In chapter 3, we dive into the Kitti Dataset. This chapter outlines the dataset used for training and testing the proposed model. The chapter includes an in-depth explanation of the KITTI dataset, including the classes (Car, Pedestrian, Cyclist, etc.) and the pre-processing techniques applied to the data, such as point cloud standardization and occlusion augmentation.

In chapter 4, we explain the methodology and the proposed approach in detail. It discusses the two-stage pipeline for 3D object detection, including the generation of 3D proposals from point clouds and the fusion architecture of GLENetVR and SE SSD for object detection. The chapter also describes the model fusion strategies and the architecture of each compents of the proposed model.

In chapter 5, we present the results of the proposed model and compares its performance with state-of-the-art methods like SE SSD, GLENetVR, and VirConv-S. Detailed analysis of performance metrics such as accuracy, precision, recall, F1 score, and AUC is provided. The computational efficiency of the proposed model is discussed in comparison to the other models, along with insights into the strengths and limitations of the approach.

Finally, in chapter 6, we draw the conclusion of the study we conducted.

Chapter 2

Literature Review

In the studies of 3D object detection techniques, it is often leverage on point cloud data, captured through sensors such as LiDAR, to provide spatial and geometric insights, which are crucial for reliable scene understanding. The challenge lies in processing this unstructured and sparse data efficiently. This paves the way for innovative research and development in this domain.

2.1 Point Cloud based 3D object detection

Point cloud-based 3D object recognition represents an important frontier in computer vision, motivated by potential applications in autonomous navigation, robotics, and augmented reality but presenting unique challenges with irregular data and sparsity of the data. To address these challenges, researchers have explored various approaches, including grid-based and point-based approaches, each with its strengths and limitations. Grid-based methods have been used to convert point clouds into regular 2D scenes or 3D voxels [17]. This research shows that, although this approach brings structure to the data, it introduces quantization loss, where the continuous properties of the point clouds are discrete as a mesh. This loss can affect the accuracy of object recognition, especially in cases of geometric accuracy is important. On the other hand, point-based methods, exemplified by techniques such as PointNet, learn from individual points directly in the cloud. This approach may face the challenges of capturing complete semantic information from the data while avoiding quantization loss. PointNet's ability to sense complex relationships between points may be limited, potentially resulting in incomplete or incorrect identifications. In response to these challenges, the proposed gateway attention-based point-set abstraction (GAPSA) algorithm appears as a promising solution [17]. GAPSA represents a new approach that aims to bridge the gap between the geometric and semantic understanding of point clouds. The system uses cognitive techniques to highlight important parts of the data, enabling it to recognize geometric and semantic features simultaneously. The main innovation lies in GAPSA's ability to focus on specific points in the cloud, taking into account their spatial relationship and relative semantics. This sharp focus increases the completeness and accuracy of visualization. It not only addresses issues of irregularities and irregularities but also provides a solid foundation for understanding complex point cloud data. By combining geometric and semantic insights, GAPSA provides a holistic approach to meaningful cloud-based 3D object recognition. It also promises that point-based

logical differentiation will overcome the limitations of network-based quantization, opening the way for accurate and reliable object detection in complex and dynamic environments. As researchers refine and extend GAPSA, its potential applications become more apparent, from improving the reserve capabilities of autonomous vehicles to providing augmented reality experiences.

Traditional 3D object recognition algorithms face challenges when objects lack complete shape information due to distance or occlusion. The proposed algorithm in [21] addresses this issue by exploiting the capabilities of Ada-GRU (adaptive gated recurrent unit) which is a type of Recurrent neural network (RNN). Ada-GRU plays an important role in the algorithm by seamlessly merging features extracted from each frame with hidden features from the previous frames. This dynamic fusion mechanism enhances feature detection, period suffering from incomplete design issues due to factors such as distance or obstruction. Another important aspect of the algorithm is how uncertainty is handled, especially for peripheral and blocked objects. Acknowledging that estimating the location of such objects can be challenging. This paper [21] introduces a probability distribution model based on a Gaussian distribution function This model, together with the corresponding bounding box loss function, is presented as this algorithm which can detect and estimate the uncertainty, associated with bounding box positioning with immense accuracy. The outstanding achievement of this algorithm is that it achieves this improvement without significantly increasing the complexity of the algorithm. This is critical for real-world applications, especially in autonomous driving, where computational effort is paramount. As such, the algorithm provides a dynamic and robust solution for detecting objects in a 3D point cloud, which is key to enabling autonomous vehicles to sense and navigate the environment. Incorporating timeline information and dealing with uncertainty, enhances informed decision-making by vehicles. The experimental results presented in the paper [21] provide strong evidence for the effectiveness of the algorithm. Detection accuracy is greatly improved, especially in situations involving distant or obstructed objects. These developments have the potential to move the autonomous vehicle industry forward, enabling it to better meet real-world challenges and pave the way for safer and more efficient autonomous vehicles on the roads. As such, the algorithm provides a dynamic and robust solution for detecting objects in a 3D point cloud, which is key to enabling autonomous vehicles to sense and navigate the environment By incorporating timeline information and dealing with uncertainty, it enhances informed decision-making by vehicles.

Another research [5] introduces Part-A2Net, a new 3D object recognition system, which marks a significant step forward in this field. The Part-A2Net process consists of two important phases: a fraction-aware phase and a fraction-gathering phase. These techniques work in concert to improve the accuracy and reliability of 3D object recognition from point cloud data. In the awareness phase, the system adopts a different perspective. It uses free-side views derived from 3D ground-truth boxes. These observations are used to predict high-level 3D coordinates and determine the precise locations of object components. This phase plays an important role in introducing preliminary concepts and establishing precise component locations, laying a strong foundation for subsequent applications. The part-accumulation stage takes the known part regions of the object and reconstructs the location of the object

by analyzing the spatial relationship between these parts. This stage increases the accuracy of the object location by capturing and adding the geometric details of the object parts to the scoring system. A notable achievement of the Part-A2Net framework is its ability to achieve state-of-the-art results on benchmark datasets such as KITTI 3D object detection using only LiDAR point cloud data. This demonstration highlights how the framework is effective in dealing with the challenges of 3D objects found in real-world situations. The paper [5] also builds on previous work, in particular extending the original PointRCNN method to the Part-A2Net framework. This extension further enhances the performance of 3D features detected from point cloud data. Notably, the system avoids the use of highly pre-defined 3D anchor boxes, which simplifies the process and constrains the 3D shapes to be imported only from facial regions, thus improving performance. In summary, the Part-A2Net framework represents a significant advance in point cloud-based 3D object recognition. Its innovative phase detection, spatial relationships, and free reference functions help increase accuracy and efficiency and overcome common challenges to maximize the power of state-of-the-art results.

2.2 Related Works

L. Wang et al. [20] proposes a 3D vehicle recognition network based on images and point clouds. It includes the first fusion module, the BEV encoding format, and the Feature Fusion (2F) network. The proposed solution uses a first-class fusion method to combine data from LiDAR and camera sensors, enabling smoother programming and improving convolution coding effectiveness. This method uses a color point cloud a bird's eye perspective representation is used to detect vehicle speed. However, it states that one of the main challenges in 3D scene perception is that the 3D data consists of a large field of view (FOV) of irregular and unstructured points, which requires consideration of data representation and arrangement suitable for CNN design in detail. The findings show that the proposed method improved the detection of blocked and remote vehicles and can finally be evaluated by end-to-end training. The method proposed in the KITTI benchmark provides more accurate real-time execution.

Another paper [13] describes a novel approach for vehicle detection from point clouds and images using a multilayer fusion network. The proposed method has the potential to improve vehicle safety by providing accurate and efficient vehicle recognition. The author's approach is to use a multilayer fusion network for 3D vehicle recognition from point clouds and images. This fusion has three forms: data-level fusion, feature-level fusion, and deep fusion. In the data-level fusion step, the network provides points with rough texture information from the RGB images of the first fusion module. Then, point clouds are encoded in both voxel grid and Bird's Eye View (BEV) formats, their abstract features are extracted and fused to output proposals using a new coarse-fine detection header with greater recall. The search header simulates a two-stage detection network to obtain coarse proposals at the encoder and optimize them at the decoder. Finally, the deep fusion module improves the reliability of quality samples by re-fusing image components, thus reducing false detection. The experimental results show that the proposed method is effective for

object recognition accuracy and can reduce the incorrect detection of blocked and peripheral objects, as well as the detection of objects with similar shapes lies so well. Thus, the paper identifies two research gaps in traffic detection. The first difference is the lack of visibility of obstructions and peripherals, which is a common problem in traffic areas. The second difference is the false detection of objects of the same shape, which can also be a serious problem in automotive safety. The paper argues that this particular problem needs to be further studied and addressed to improve accuracy.

Another research [16] proposes a LiDAR-camera-based fusion algorithm to improve the trade-off between dense semantic information from the camera and accurate depth information from LiDAR for real-time object detection in autonomous driving. The algorithm converts unprocessed point clouds into camera planes producing 2D depth images using the depth and the RGB function. These branches are connected by a cross-feature fusion block. Data fusion is also performed by feature-level fusion technique. The proposed method outperforms other state-of-the-art algorithms and provides more performance and real-time efficiency at different complexity levels according to experimental results on the KITTII dataset. Wire detection and conceptual strategies are introduced to improve mediation focus on areas of interest and to reduce false negatives and false positives.

3D object recognition using point clouds is complicated by its incomplete shape and weakness of point clouds. H. Liu et al. [11] discusses the challenges of accurate 3D object recognition of point clouds and suggests that another two-dimensional Deformable Pyramid R-CNN scheme is not developed. For multilevel selection of 3D features based on the sparsity of non-empty voxels in a region of interest (RoI), it also presents a voxel feature pyramid. Deformable Voxel RoI Pooling, a method that KITTI provides a coherent definition for accuracy for identifying by abstracting rich contextual information from voxels of interest beyond the RoI. For vehicle identification in the dataset, the method outperforms PV-RCNN 0.47%, 1.63%, and 1.34% in terms of it being in the mild, medium, and hard range. The method achieves consistent performance on both the KITTI Dataset and the Waymo Open Dataset.

In another research [7], the authors propose a LIDAR-based approach for object detection in 3D using a fully flexible convolutional network (FS 2 3D) and foreground segmentation. Sparse enhancement detection heads are used to predict the target and the bounding box at each active point in the sparse feature map, which transforms the search problem into a bird's eye classification problem. A new bounding box coding technique and associated loss functions were developed. For cars and motorcycles, the technique surpasses the most advanced LIDAR-based solutions in terms of speed and accuracy. Compared to the dense backbone mesh, the sparse convolutional backbone mesh presented in the paper is 2.2 times faster and uses 18.4 times fewer FLOPs. To predict objects 3D bounding box, the study uses multitasking mesh design results in a detecting head with three branches. The addition of a new boundary box reference registration method and associated loss in birds-eye view (BEV) and 3D search enhances the performance of the loss function based on the boundary box rule of 1.1% and 0.8%.

The study [14] proposes a multilayer fusion network for 3D vehicle detection, which enhances the blocking and remote object performance and reduces the false detection of equal-sized objects. A first fusion module is provided for data-level fusion of images and point clouds, unlike the existing fusion-based techniques. Navat - Information obtained The study proposes a new coarse-fine detection header for traffic scene characterization, using 3D points in point clouds as voxel meshes and BEVs, resulting in more accurate representation and semantic information Improves the accuracy of our search As to evaluate results, their method outperforms many SOTA algorithms, especially for objects a of blocked and peripheral. It also reduces the blurring of objects with similar shapes. Through ablation experiments, the recommended modules of the assay are successful in improving the detection performance. The complex network topology of the analysis results in lower real-time performance compared to previous methods, and further research is needed on a smaller network sample.

L. Wiesmann et al. [8], discusses a different approach based on neural networks for loss point cloud compression was developed. Their approach relies on complex and large-scale maps constructed from integrated point clouds mounted in autonomous vehicles. Their method uses a deep convolutional autoencoder to identify a small collection of feature descriptors using the common schemes. The descriptor set acts as a compressed representation, which the decoder can use for later point cloud reconstruction or efficient storage transmission. Furthermore, to avoid memory space issues from skip connections or discretization effects from using voxel grids practically, the research proposes that 3D deconvolution acting directly on points can be successfully reduced.

In another research [3] surveys, the rapid development of deep learning-based 3D object recognition technologies has been facilitated by advanced computational tools. However, more accurate real-time methods are being sought. Research focuses on developing new deep-learning architectures that extract finer features and provide new data representations aimed at faster processing for more accurate and efficient detectors. The addition of a second detector enhanced post-processing NMS. The challenges of automatic 3D object recognition include negative and positive unbalanced samples and complex sensitivity conditions, which need further development to improve classification reliability and local accuracy This study compares different 3D object detection techniques and provides a comprehensive literature review. It aims to provide readers with valuable information about the 3D sensitivity of point clouds and help in research questions. The proposal proposes a general 3D object detection system, a framework for analyzing key sample features, and a comparative analysis of state-of-the-art methods The study also addresses issues related to methods based on deep learning for LiDAR point cloud processing and suggests possible future research methods.

Also, another study [10] investigates a point-based 3D object detection method in a 3D object detection algorithm. This study investigates PointRCNN—a 3D object recognition system using deep learning. Third-order point cloud classification and image classification are presented to increase the classification confidence. The first proposed point cloud classifier and image classifier are introduced with their mesh

structure and how to return and resolve the objective. Then, the network design and loss function of the proposed point cloud classifier and image classifier are described in detail. Finally, model tests are conducted to demonstrate that it can achieve a high sensitivity of 79.51% (moderate). The accuracy of the model is 0.66% better than the original, demonstrating the robustness of point clouds and third-order image classification. In the KITTI validation set, the 3D detection results of the model in this study are compared with the traditional 3D target detection method. In this study, if the 3D intersection ratio between the detection frame and the label exceeds 0.7%, the detection frame is considered correctly detected; Otherwise, the search is considered unsuccessful. Table 1 shows the detection results obtained by the model in the validation set using the mentioned methods. As can be seen, this method has a higher detection rate than the original 3D target detection algorithm, but a faster speed than the original 3D target detection algorithm, which has been reduced. This is because the method presented in this study is point-based, and each of the three steps requires multiple inputs, mask operations, configuration changes, etc. Although it has a higher time complexity.

Chapter 3

Dataset Description

3.1 Data Collection

Point cloud data is collected from LiDars. The Point clouds are collected from different types of urban environments. The Point cloud datasets are released to check the benchmark of different 3D object detection models. The three most competent datasets are the Kitti Dtaset, The NUplan Dataset, and the ApolloScape Dataset. We have decided to use Kitti Dataset for it's availability and world-wide credibility in case of Model Evaluation.

3.1.1 NUplan Dataset

The world's first ML planning benchmark, 1200h of driving data from 4 cities (Boston, Pittsburgh, Las Vegas and Singapore)Sensor data released for 120h (5x LIDAR, 8x camera, IMU, GPS), 5B 3D bounding boxes auto labeled for 7 classes.

3.1.2 ApolloScape Dataset

Trajectory dataset, 3D Perception Lidar Object Detection and Tracking dataset including about 100K image frames, 80k lidar point cloud and 1000km trajectories for urban traffic. The dataset consisting of varying conditions and traffic densities which includes many challenging scenarios where vehicles, bicycles, and pedestrians move among one another.

3.1.3 Kitti Dataset

The 3D object detection benchmark consists of 7481 training images and 7518 test images as well as the corresponding point clouds, comprising a total of 80,256 labeled objects. This Dataset is widely used in computer vision and autonomous driving research. It contains a variety of scenes captured from a car equipped with LiDAR sensors and cameras, making it ideal for tasks like 3D object detection, tracking, and segmentation. There are three kinds of data in this Dataset: Easy, Moderate, Hard. We used the Moderate order. From our analysis, the dataset contains the following classes: Car, Pedestrian, Cyclist, Van, Person sitting, Truck, Tram, Misc. These classes encompass various common objects found in roads in the cities, particularly focusing on road traffic scenarios.

Pickle File Creation: The pickle files in the Kitti dataset is created to store processed data in a serialized format.
Data Collection: Raw data is collected from the sensors, which may include point cloud data from LiDAR, images from cameras, and annotations (like bounding boxes and class labels) for each object in the scene.
Data Processing: The raw data is processed to extract useful features which involves: Filtering the point cloud data to remove noise, Projecting LiDAR points into the camera images to associate 3D data with 2D images, and Annotating objects with bounding boxes and class labels.
Serialization: The processed data, including features and annotations, is serialized into a binary format. This is done using Python's pickle library. This allows for efficient storage and quick loading of large datasets.
Saving: The serialized data is saved to .pkl files, which can be easily loaded into Python programs for further analysis or training machine learning models.

Figure 3.1: Raw Image Data from Kitti Dataset

Figure 3.2: Bounding Box Data from Kitti Dataset

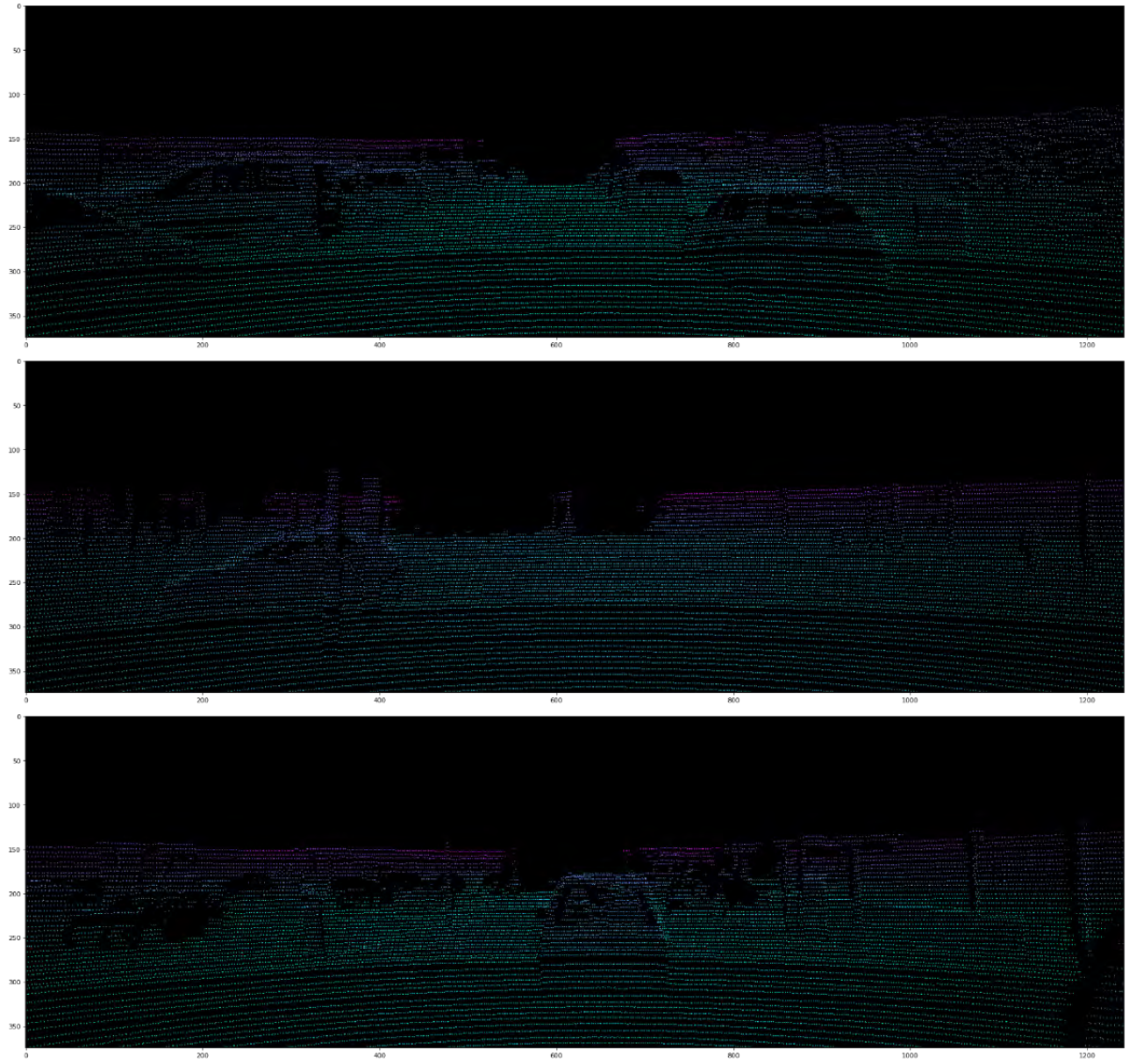

Figure 3.3: Point Cloud Data from Kitti Dataset using Color Map

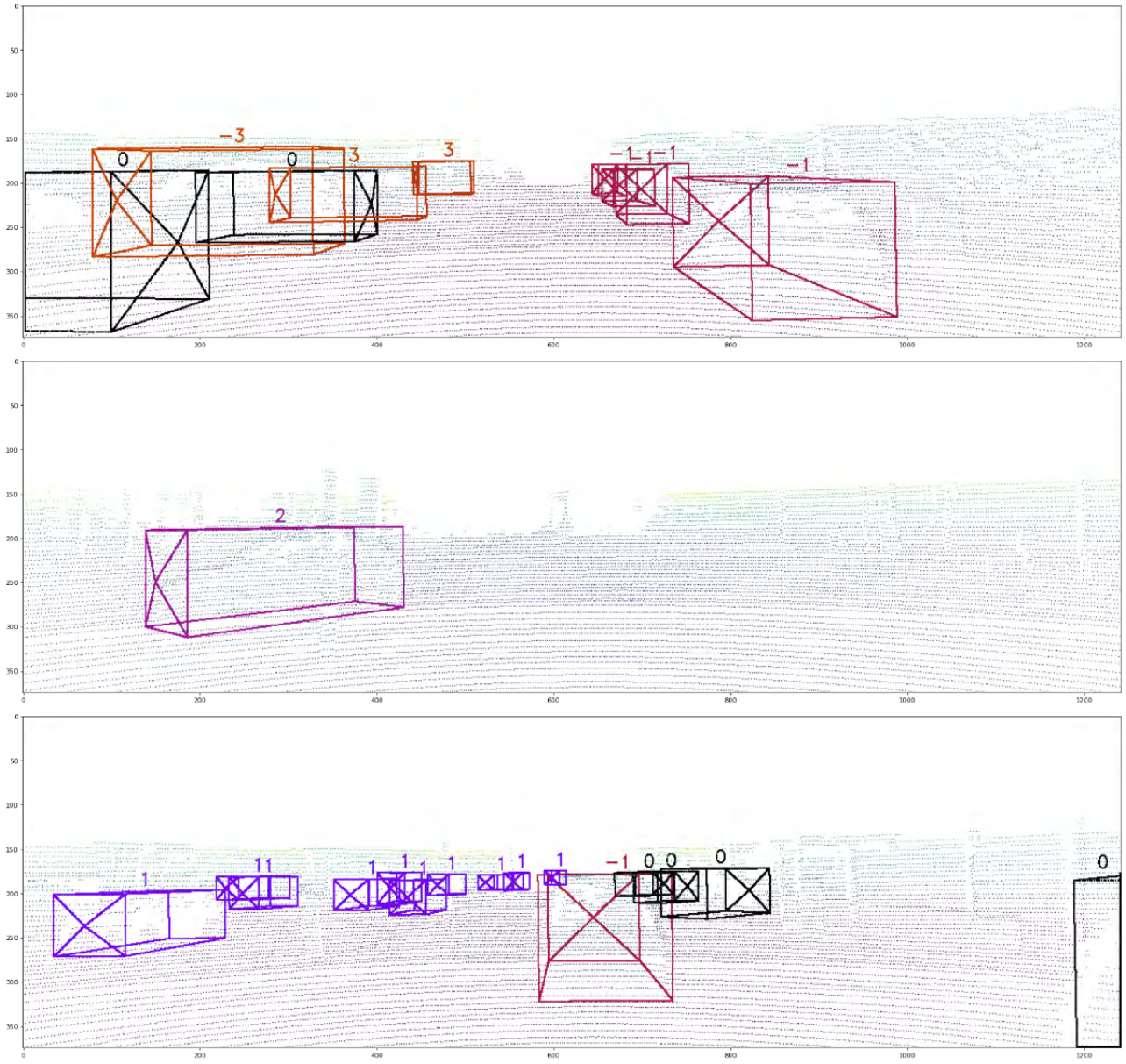

Figure 3.4: Full Data from Kitti Dataset

3.1.4 Data Preprocessing

3.1.4.1 SE-SSD Preprocessing

These are the techniques used in SE-SSD model for Pre processing. Data Normalization: Standardization: This involves scaling the point cloud data to have zero mean and unit variance. This step helps stabilize the training process and improves the convergence rate of the model.
Voxelization: The LiDAR point clouds are often voxelized into a structured grid. Voxelization involves dividing the 3D space into a grid and point cloud data into the grid cells. This reduces the number of points and simplifies the data structure.
Random Occlusion: Simulating occlusions by randomly removing some points from the point cloud helps the model generalize better. This mimics real-world scenarios where objects can be partially obscured.
Random Rotations and Translations: This technique increases the efficiency of the model by training it on different orientations and positions of the objects.
Data Augmentation: Applying transformations like flipping, rotation, and scaling to increase the diversity of the training dataset.
Input Scaling: Rescaling the coordinates of point clouds to a certain range (e.g., [-1, 1]) ensures that the data is on a consistent scale, which is crucial for neural networks.

3.1.4.2 $GLENet - VR$ Preprocessing

Point Cloud Encoding: In $GLENet - VR$, point clouds are encoded into feature representations that capture spatial information effectively.
Local and Global Feature Augmentation: $GLENet - VR$ employs local feature extraction to capture fine details of the object.
Augmentation Techniques:Similar to SE SSD, $GLENet - VR$ use various data augmentation techniques, including random rotations, translations, and possibly synthetic occlusion, to create robust training samples.
Standardization: Like SE SSD, $GLENet - VR$ benefits from standardizing point cloud data to improve model training.

3.1.4.3 Preprocessing Technique used for the Modified Model

We used Occluded Point Cloud Data and Standardized Point Cloud Data. The other techniques are also followed, however we get the best output and efficient computational power using these two methods. For our limited computation power we took 7,481 items from the Moderate Dataset. The Dataset was split into trainning, testing and validation folders respectively in the ratio of 70:20:10.
We used color maps and visual class to visualize data from the Kitti Dataset.

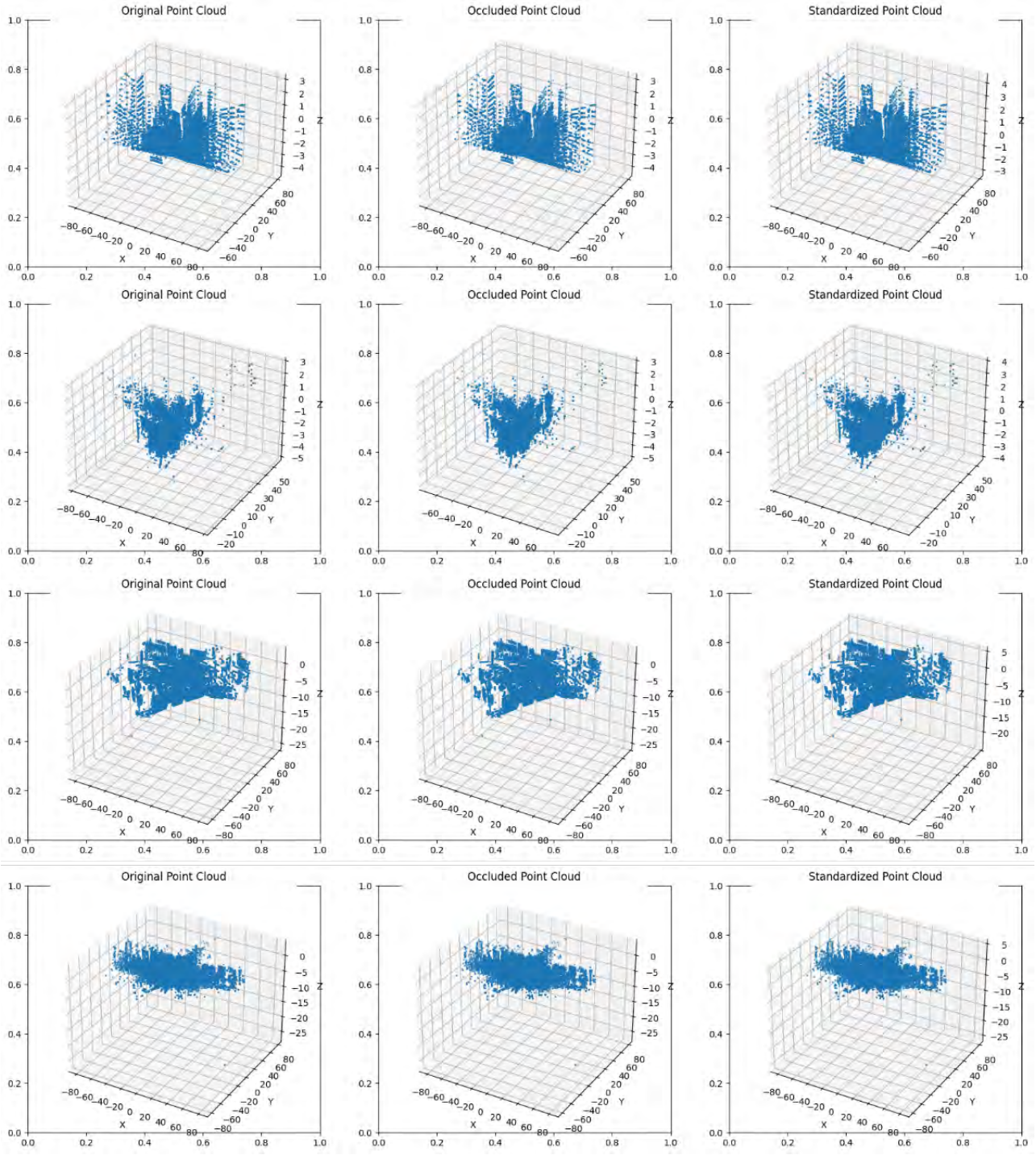

Figure 3.5: Pre Processed Data

Chapter 4

Methodology

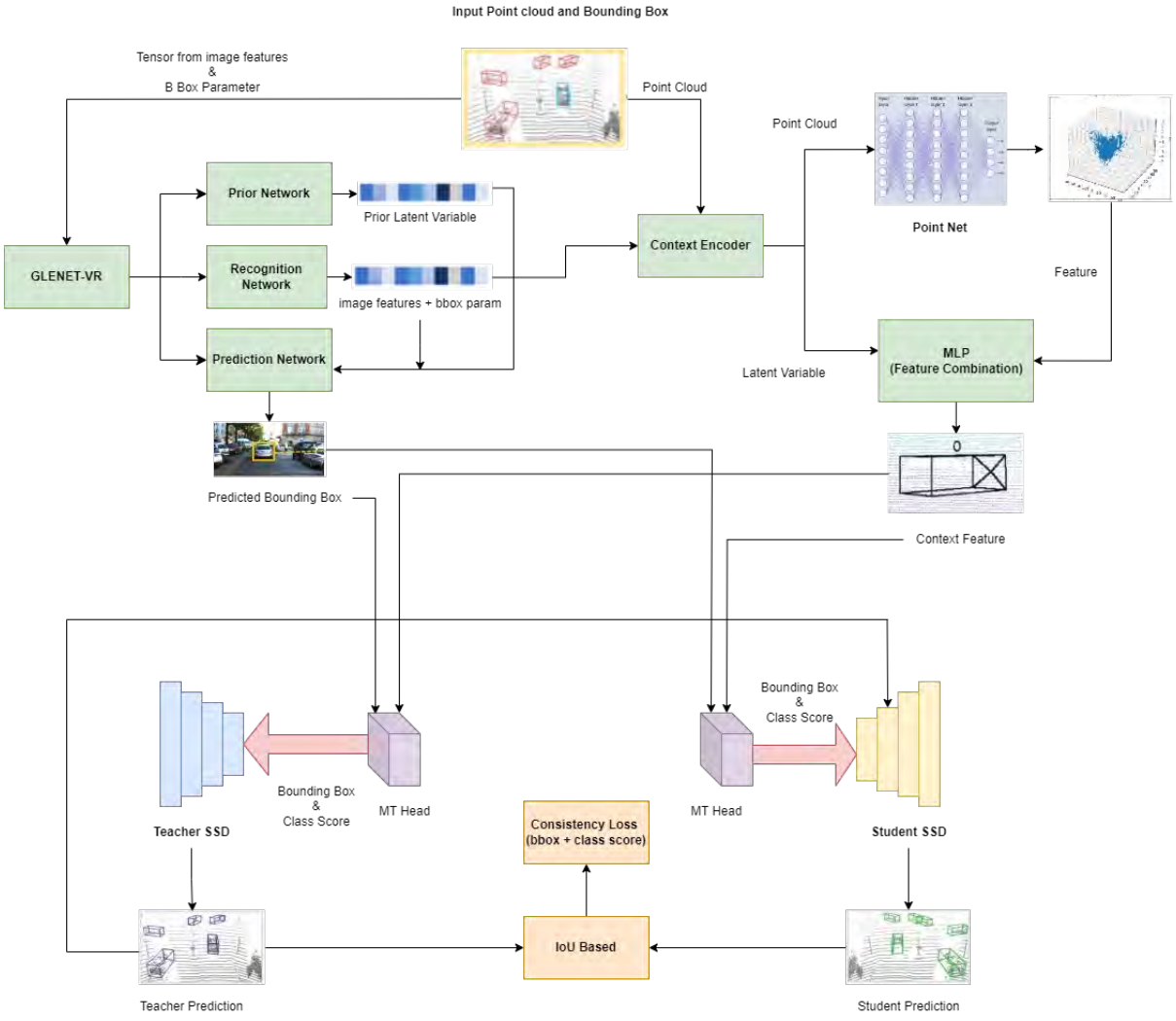

Figure 4.1: The top level overview of the propose GLE-SSD-VR.

4.1 GLE-SSD-VR Model

- Model Fusion Framework: Developing a fusion framework that combines the strengths of $GLENet-VR$-VR and SE-SSD for robust 3D object recognition using LiDAR datasets.We will leverage $GLENet-VR$'s uncertainty-aware quality estimator architectures to enhance the training of SE-SSD's Intersection over Union (IoU) branch, ensuring effective utilization of predicted localization uncertainty.

- Conditional Variation Autoencoders (VAEs) Integration: We will implement a seamless integration of $GLENet-VR$'s conditional VAEs into the SE-SSD architecture, enabling the generation of robust 3D object representations from fuzzy annotations. Quantify label uncertainty introduced by $GLENet-VR$ and incorporate it into SE-SSD, transforming it into a probabilistic model that better understands and represents uncertainty in object localization during both training and inference.

- Practical Implementation and Performance Evaluation: We will implement the integrated $GLENet-VR$-VR and SE-SSD framework into popular 3D detectors, ensuring practical applicability. We will evaluate the performance of the integrated model on benchmark datasets such as KITTI and Waymo, demonstrating improved accuracy in 3D object recognition, particularly surpassing previous Lidar-based methods and excelling on challenging datasets like KITTI.

In the beginning the models takes Point Cloud and Bounding Box as input. The Tensor image features along with Bounding Box Parameter goes to $GLENet-VR$ as showed in figure 4.1. In the Prior Network of $GLENet-VR$ the image features is made to latent variables. The Recognition Network of $GLENet-VR$ uses both image features and Bounding Box Parameters to make featured Latent variables. This Recognition Latent Variables gets fed to Context Encoder later. In the Prediction Network of $GLENet-VR$ the latent variables gets fed and generate Predicted Bounding Box as showed in figure 4.1. In the Context Encoder class, PointNet generates Feature which is fed to MLP. In MLP latent variable from $GLENet-VR$ combines with the feature got from PointNet to generate Context Features. The context features from Context Encoder and Predicted Bounding Boxes from $GLENet-VR$ get fed to $MTHead$ of the both Teacher SSD and Student SSD. The Teacher SSD is complex, thus gives target predictions as "ground truth" to the Student SSD. We get IoU of the predictions of Teacher SSD and Student SSD. This is used to calculate Consistency Loss of Bounding Box and Classification. The calculated loss is then back propagates to Student SSD for knowledge distillation as showed in figure 4.1.

4.2 Description of GLENet-VR

Point cloud-based 3D object recognition represents an important field in computer vision, especially for autonomous vehicles and robotics. In the study, [22] addresses an important challenge in this area: the uncertainty of the ground-truth description of 3D bounding boxes, which can lead to confusion in the deep training of 3D object

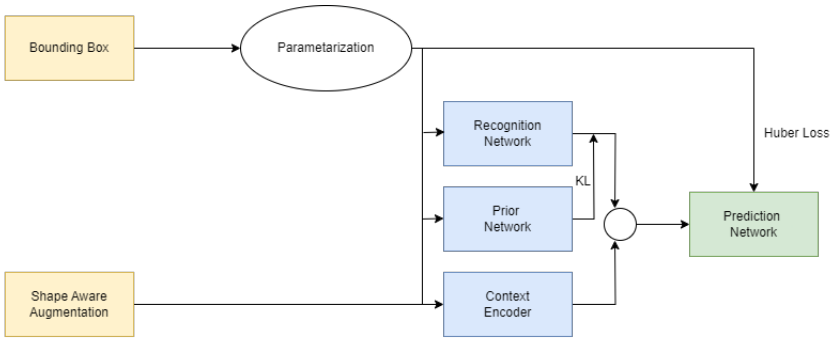

Figure 4.2: The Architecture of $GLENet - VR$

detector's intensity and reduce eventual detection accuracy. To address this issue, the authors introduce a novel framework called $GLENet - VR$. $GLENet - VR$, a short generation class for 3D object recognition from fuzzy annotations, is a generation algorithm based on conditional variation autoencoders (VAEs) with the main objective of being generally robust to 3D objects and their downstream capabilities true boundary boxes between One and many relationships are modeled. This correlation is due to the inherent uncertainty in defining object boundaries in 3D space. $GLENet - VR$ addresses this by introducing label uncertainty, effectively quantifying ambiguity in ground-truth identification. This label uncertainty can then be easily incorporated into existing 3D object detectors for depth objects. In doing so, it transforms these detectors into probabilistic models, increasing their ability to understand and represent uncertainty in object localization during training and inference The main feature of the paper [22] is the introduction of uncertainty-aware quality estimator architectures. This algorithm plays an important role in guiding the training of the Intersection over Union (IoU) branch by exploiting the predicted localization uncertainty In particular, it ensures that the training process uses ambiguous knowledge and calculates it, resulting in more accurate and robust detections. Importantly, the proposed methods are not only theoretically explored but also practically implemented by integrating them into popular base 3D detectors. The results are impressive, showing performance greatly benefits from benchmark datasets such as KITTI and Waymo. $GLENet-VR$ in particular stands out as outperforming all previously published LiDAR-based methods and outperforming single methods on the complex KITTI dataset. The contribution of this paper is noteworthy, as they directly address a key issue in point cloud-based 3D object recognition: annotation ambiguity By introducing $GLENet - VR$ and related methods, the authors provide tools that account for this ambiguity and resolving the field, thereby increasing detection accuracy and self-sustaining systems based on the accuracy of

3D object recognition such as self-driving cars and robot motion.

4.3 Description of SE-SSD

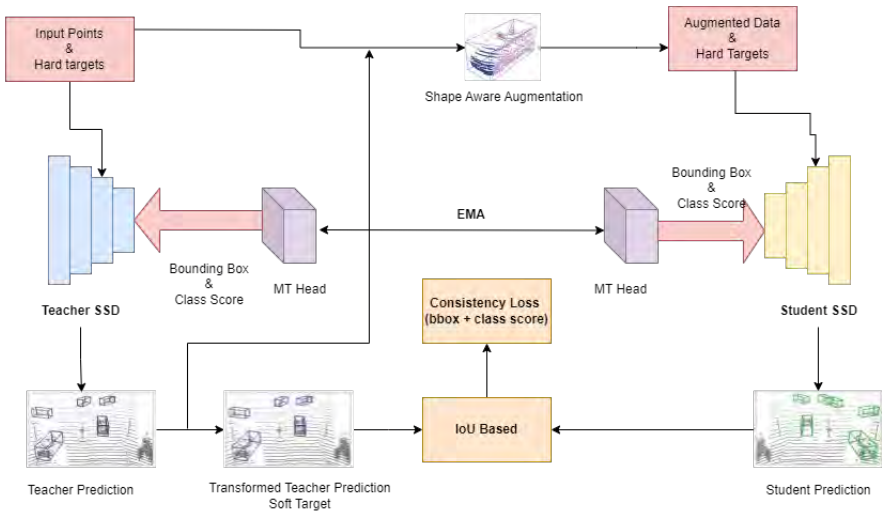

Figure 4.3: The Architecture of SE-SSD

A recent study [9] introduces a new framework that represents a breakthrough in this field which is called entitled SE-SSD: Self-Ensembling Single-Stage Object Detector From Point Cloud.
The SE-SSD architecture is designed to provide accurate and efficient 3D object detection in external point cloud data. It makes several key contributions, each contributing to the effectiveness of the whole. At the core of the SE-SSD is a self-assembling device, which is optimized by limiting stabilization with flexible values. This approach allows the system to iteratively adjust and improve its predictions during training. By exploiting the knowledge of a simple objective, SE-SSD can fine-tune its predictions, resulting in more accurate and reliable detections. Furthermore, the paper [9] introduces ODIoU loss which plays an important role in managing networks with complex objectives. This loss function contributes to visual accuracy by providing a strong observer signal as shown in Figure 4.2. It helps the network learn how to construct bounding boxes that are more consistent with

the ground truth, ultimately improving detection performance. Another distinctive feature of the SE-SSD architecture is its size-aware data enhancement strategy. This strategy aims to increase the diversity of training models, which is often necessary for complex and comprehensive learning. By introducing data sets, the network is equipped to deal with real-world complexities and nuances. Perhaps the strongest aspect of the SE-SSD is its functionality. It outperforms all state-of-the-art 3D and birds-eye view (BEV) vehicle detection methods at the kitty scale. This improvement highlights the effectiveness of the system and highlights its potential to set a new standard in 3D object recognition. Furthermore, the SE-SSD does not sacrifice performance for accuracy. It achieves very high computational speed, making it suitable for real-time applications where speed and accuracy are paramount. In conclusion, SE-SSD represents a remarkable breakthrough in point cloud-based 3D object recognition. Its self-assembly innovation, missing ODIoU implementation, and highly data-aware scalability combine to contribute to exceptional performance. As autonomous systems and robotic applications continue to evolve, SE -SSD remains a promising tool to increase holding power, ultimately helping to sail safely and effectively in challenging real-world conditions.

4.4 Working Process of GLE-SSD-VR

The workflow begins with the input of point cloud data, which represents 3D spatial information. This data consists geometric details, allowing the model to understand the physical layout of objects within a scene. The point cloud consists of discrete data points, each characterized by its coordinates (x, y, z) in a three-dimensional space, along with additional attributes such as color. This geometric data format provides spatial cues which helps object localization and classification. Alongside the point cloud data, image data gives visual information that helps the spatial characteristics of the point cloud. Images provide contextual details, such as texture, color, and visual patterns, which are important for distinguishing between objects that may have similar geometric shapes but differ in appearance.

4.4.1 Context Encoder

The point cloud data is first processed by the Context Encoder, which is a class designed to extract meaningful features from the point cloud. The Context Encoder includes a PointNet layer that captures local features of the 3D data while preserving global contextual information. Following this, the features are combined with latent variables that represent contextual knowledge or noise factors. This combination creates the feature representation by introducing relevant information that might not be present in the point cloud data.
After the Context Encoder processes the point cloud, the next stage involves passing these enriched feature representations to the $MTHead$. This head is responsible for leveraging the combined features of both the point cloud and image data.

4.4.2 $GLENet - VR$ Integration

The image data is fed into the $GLENet-VR$, which is a class made to extract visual features. This network is built to analyze the RGB images, detecting important

attributes and relationships within the visual context. The outputs from $GLENet-VR$ provide high-level visual features that help the spatial features obtained from the point cloud.

4.4.3 Combining Outputs for Detection

The architecture leads to integration of outputs from both the Context Encoder and $MTHead$. These combined features are essential for the $MTHead$, where they are utilized in a multi-task learning framework. The $MTHead$ have two separate networks: the Teacher SSD and the Student SSD. The Teacher SSD acts as a guide, providing target predictions based on the combined feature inputs. In contrast, the Student SSD learns from the Teacher's predictions, allowing it to develop similar accuracy while optimizing for computational efficiency.
This architecture not only facilitates effective object detection by synthesizing spatial and visual information but also ensures that the model can generalize different kind of environments. By leveraging knowledge distillation, the workflow enhances the Student SSD's performance while maintaining an efficient model for real-time applications. The result is a powerful and efficient 3D object detection system capable of accurately detecting and localizing objects in complex environments.

4.5 Model Architecture

4.5.1 Description of $GLENet-VR$

The $GLENet-VR$ (Global and Local Enhanced Network for Visual Recognition) plays a crucial role in processing 2D image data to generate visual feature maps that contribute to the overall performance of the multi-modal architecture for 3D object detection. By employing a hierarchical structure made with three major components—the Prior Network, Recognition Network, and Prediction Network. $GLENet-VR$ is designed to extract a rich representation of the image data, facilitating accurate object detection.

4.5.1.1 Prior Network

The Prior Network serves as the initial stage of $GLENet-VR$, where raw image data is introduced. This network works to preliminary feature extraction, focusing on capture low-level features present in the image.
Fully Connected Layers: The Prior Network consists of four fully connected layers: Layer 1: Linear(512, 64), Layer 2: Linear(64, 128), Layer 3: Linear(128, 512), Layer 4: Linear(512, 8) (Output latent variable dimension)
Layers: Input image features (dimension: 512) are passed through the first fully connected layer, generating 64 features. The output from Layer 1 is processed through Layer 2, producing 128 features. This is followed by Layer 3, which outputs 512 features. Finally, Layer 4 produces an 8-dimensional latent variable, which represents essential low-level features.
Feature Extraction Techniques:
Activation Function: The $ReLU$ activation function is applied after each fully connected layer, introducing non-linearity and enabling the network to learn complex

patterns by setting all negative values to zero.
Batch Normalization: This technique is typically used in conjunction with the fully connected layers to stabilize and accelerate the training process.
In summary, the Prior Network effectively lays the groundwork for feature extraction by capturing low-level image characteristics, ensuring that subsequent networks can build upon a rich and nuanced representation of the input data.

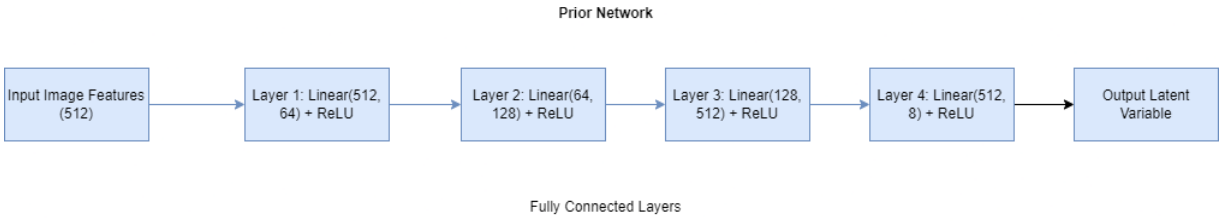

Figure 4.4: Layers of Prior Network

4.5.1.2 Recognition Network

Building upon the foundational work of the Prior Network, the Recognition Network works with extracted features, enabling the detection of more complex patterns such as shapes, objects, and contextual relationships within the image.
Fully Connected Layers: The Recognition Network also comprises four fully connected layers: Layer 1: Linear(512 + 7, 64) (combining prior features with 7 bounding box parameters), Layer 2: Linear(64, 128), Layer 3: Linear(128, 512), Layer 4: Linear(512, 8) (Output latent variable dimension)
Data Flow:
Layer 1: Linear layer with input size 519 (512 features from the Prior Network + 7 bounding box parameters) and output size 64. This step reduces the dimensions and helps capture essential features while maintaining contextual information. Activation Function: *ReLU* is applied after this layer to introduce non-linearity, which helps the network learn complex patterns. Dropout: A dropout layer is used to randomly drop units, helping reduce over-fitting.
Layer 2: Linear layer with input size 64 and output size 128. This expands the feature space, allowing the network to model higher-level interactions. Activation Function: *ReLU* again helps activate complex patterns.
Layer 3: Linear layer with input size 128 and output size 512. This step further increases the complexity and dimensionas of the feature representations. Activation Function: *ReLU* is applied to maintain non-linearity.
Layer 4: The final linear layer maps the 512-dimensional output to an 8-dimensional latent variable. This latent variable is a condensed, learned representation of the object's features.
Techniques:
ReLU Activation: Introduces non-linearity, enabling the model to capture complex patterns.
Dropout: It is added to prevent over-fitting by randomly setting a fraction of the output units to zero during training.
The network's output is an 8-dimensional latent variable, representing the features and characteristics of the object in 3D space, learned from both the input data and the bounding box parameters.

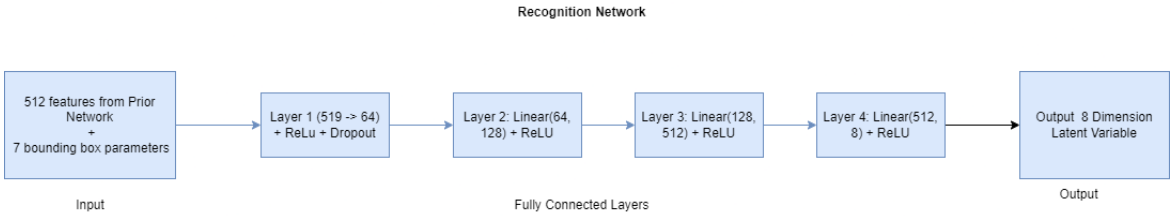

Figure 4.5: Layers of Recognition Network

4.5.1.3 Prediction Network

The Prediction Network in the $GLENet - VR$ class is responsible for predicting the bounding box parameters based on the combined latent variables from the Prior Network and Recognition Network.
Input: The input to the Prediction Network is a concatenation of two latent vectors:
Prior latent: An 8-dimensional vector produced by the Prior Network that encodes low-level features from the image.
Recognition latent: An 8-dimensional vector generated by the Recognition Network that captures both the image features and bounding box parameters. The combined vector has a total dimension of 16 (8 from prior + 8 from recognition).
Layers:
Layer 1: A fully connected layer that reduces the input dimensions from 16 to 64.
Layer 2: Another fully connected layer that maintains the same dimension of 64.
Layer 3: The final fully connected layer that outputs the bounding box parameters with a dimension of 7.
Activation Function: The *ReLU* (Rectified Linear Unit) activation function is applied after Layer 1 and Layer 2 to introduce non-linearity and help the model learn complex patterns. The output from Layer 3 is used directly to predict the bounding box parameters without an activation function, as it is a regression task.
Output: The output of the Prediction Network is a 7-dimensional vector representing the bounding box parameters for the object in the image. Bounding box parameters are 4 coordinates, 3 object classes or other features.

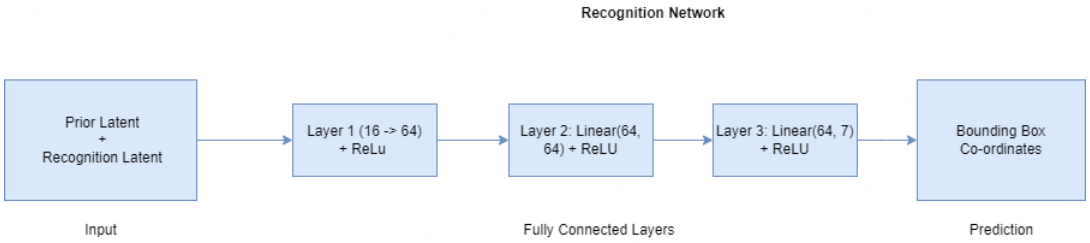

Figure 4.6: Layers of Prediction Network

4.5.2 Description of Context Encoder

The Context Encoder handles the point cloud data and latent variables, and outputs a feature representation that merges the spatial information with the latent context. This class is designed to handle the processing of 3D point cloud data using PointNet,

and then integrates these point features with latent variables using a Multi-Layer Perceptron (MLP).

4.5.2.1 Input from $GLEnetVR$ Component:

The $GLEnetVR$ model is used to process and learn key features from point cloud data in relation to the bounding box parameters. This helps as a pre-processing step where raw point cloud inputs are there into a more informative feature space. The output from Recognition Network of $GLEnetVR$ is used as input to the subsequent PointNet component.

4.5.2.2 PointNet:

Input: After Recognition Network of $GLEnetVR$ processes the data, the PointNet module receives the point cloud data as input. PointNet's main role is to handle the geometric structure of the point clouds by applying convolutional layers.

Convolutional Layers: PointNet starts by applying two 1D Convolutional layers: The first convolution has a filter size of 64 and is followed by a $ReLU$ activation function. This layer applies 1D convolution across the point cloud data. Each point is processed individually, and local features are extracted for each point. It has 64 filters.
The second convolution has a filter size of 128, also followed by a $ReLU$ activation. A second 1D convolutional layer is applied to further refine and expand the local features of each point. It has 128 filters.
These convolutional layers works to process the spatial features of each point in the point cloud individually. This helps the model to capture local features for each point.
Max Pooling: After feature extraction through convolutions, max pooling is applied across the entire set of points, which sums local features into a global feature vector. This pooling layer ensures that the model hold a global representation of the entire 3D structure. It focuses on the most prominent features across all points in the cloud.

4.5.2.3 MLP

Combining Features: After max pooling, the global features extracted by PointNet are concatenated with the latent variables. These latent variables are repeated across the dimension to match the size of the global feature vector. It ensures that both types of data are aligned.
Feed-forward Layers:
The combined feature vector is then passed through a two-layer MLP. The first layer reduces the dimensions. The global features and latent variables are concatenated and passed through a linear layer to integrate them (128 + latent variables $\rightarrow$ 128). Followed by a linear transformation followed by a $ReLU$ activation in this linear layer.
The second MLP layer further processes the features and outputs the final 128-dimensional feature vector. This integrates both the point cloud's spatial informa-

tion and the context from the latent variables.

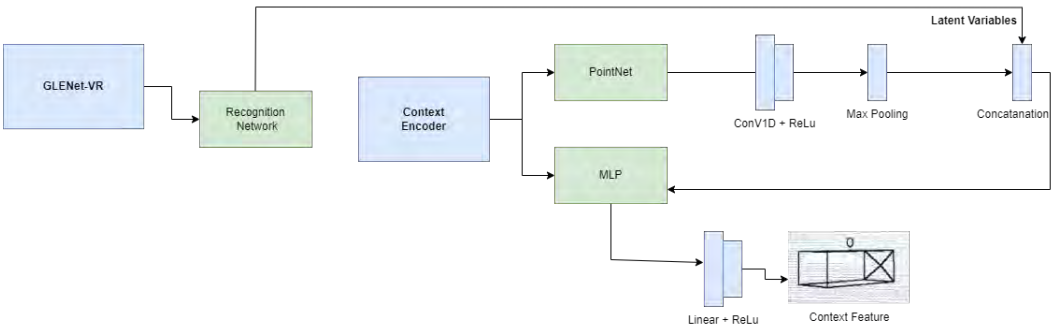

Figure 4.7: Architecture of Context Encoder Class

4.5.3 *SSD*

4.5.3.1 Teacher *SSD*

The TeacherMTHead is designed to be a more complex architecture, with a deeper and wider network than the Student version. The Teacher *SSD* consists of five fully connected layers, with progressively reducing dimensions from 512 to 64 units. This depth allows the Teacher *SSD* to capture more complex relationships and patterns from the input features. The Teacher SSD has the capacity to model intricate feature representations as it uses more layers and higher-dimensional transformations.
It has input features of 128 dimensions.
Layer Structure:
Layer 1: A fully connected layer (Linear(128, 512)) with *ReLU* activation.
Layer 2: A fully connected layer (Linear(512, 512)) with *ReLU* activation.
Layer 3: A fully connected layer (Linear(512, 256)) with *ReLU* activation.
Layer 4: A fully connected layer (Linear(256, 128)) with *ReLU* activation.
Layer 5: A fully connected layer (Linear(128, 64)) with *ReLU* activation.
After the deeper layers, the output is split into two branches:
Bounding Box Branch: A fully connected layer (Linear(64, 256)) to predict the bounding box coordinates.
Class Scores Branch: Another fully connected layer (Linear(64, 256)) to predict the object class scores.
Data Flow: Input features pass through the first layer (128 to 512) with *ReLU*. The result goes through multiple layers, reducing dimensions: 512 $\rightarrow$ 512 $\rightarrow$ 256 $\rightarrow$ 128 $\rightarrow$ 64. The final 64-dimensional output splits into two branches: one for bounding box predictions and one for class scores. The bounding box branch produces a 256-dimensional output, and the class scores branch also produces a 256-dimensional output. This architecture is designed to have capacity to capture complex patterns, making it a robust "Teacher" model in a knowledge distillation setup.

4.5.3.2 Student *SSD*

The StudentMTHead is a simpler and smaller network, which makes it faster but less complex than the Teacher model. It takes the outputs of the Teacher *SSD* as inputs. It serves the purpose of approximating the performance of the teacher but with fewer parameters, making it more efficient in terms of computation.
Description: Input Features Dimension: 128 Layer Structure: Layer 1: A fully connected layer (Linear(128, 256)) with *ReLU* activation. Layer 2: A fully connected layer (Linear(256, 128)) with *ReLU* activation. After these two layers, the output is split into two branches: Bounding Box Branch: A fully connected layer (Linear(128, 128)) to predict bounding box coordinates. Class Scores Branch: Another fully connected layer (Linear(128, 128)) to predict object class scores. Data Flow: Input features pass through the first layer (128 to 256) with *ReLU*. The result goes through the second layer (256 to 128). The 128-dimensional output splits into two branches for bounding box predictions and class scores. Both the bounding box and class scores branches produce a 128-dimensional output. This architecture is designed to be simpler and less complex, making it suitable as the "Student" model in a knowledge distillation process, where it learns from the more complex Teacher model.

4.5.4 Knowledge Distillation

4.5.4.1 Consistency Loss

Classification Consistency Loss: The Student *SSD* tries to match its class predictions to those made by the Teacher *SSD*.
Instead of just using hard labels knowledge distillation often uses the soft class probabilities produced by the Teacher. These soft probabilities carry more subtle information, such as the relative confidence the Teacher has in different classes.
Loss Function: We used KL Divergence which measures how different the Student's predicted class probabilities are from those of the Teacher. This soft-target approach is more informative than just using hard labels.
For example: If the Teacher predicts "0.6 car, 0.3 truck, 0.1 bus," and the Student predicts "0.5 car, 0.2 truck, 0.3 bus," the KL Divergence will measure how well the Student mimics the Teacher's confidence. We used Smooth L1 Loss function as well for this task. This is less sensitive to outliers which makes it a good fit for bounding box regression tasks.
Bounding Box Consistency Loss: The Student *SSD* tries to match its bounding box predictions (i.e., the coordinates of detected objects) to those of the Teacher *SSD*. The Teacher *SSD* detect an object, then the Student tries to match the predicted bounding box as closely as possible. The bounding box predictions involve both the position and size of the detected object.

4.5.4.2 Knowledge Distillation Process

Teacher *SSD* Generates Predictions: The Teacher model processes input data (e.g., images and point clouds). This produces bounding box predictions and class scores which is confidence for different object categories.
Matching Predictions (Pseudo-Labels): The function $match - targets - with -$

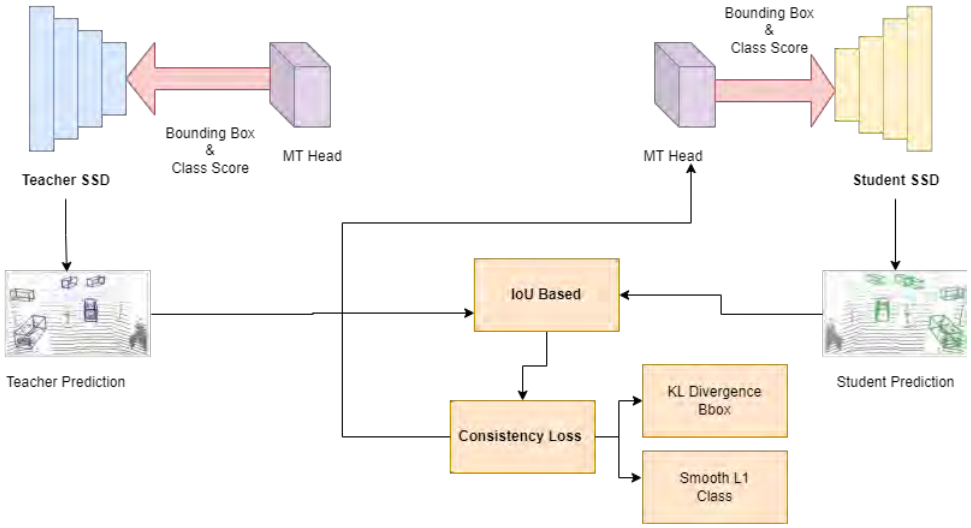

Figure 4.8: Knowledge Distillation Process

predictions aligns the Teacher's bounding box predictions with those of the Student by calculating the IoU. The IoU is calculated between corresponding bounding boxes. Matched predictions with high IoU are treated as "ground truth" for the Student during training.
Student SSD Receives Teacher Predictions: The Student SSD uses features as pseudo-labels for learning when the predictions are matched. The bounding boxes and class scores predicted by the Teacher SSD are treated as the target bounding boxes for the Student SSD.
Consistency Loss Calculation: The Consistency Loss calculates the difference between the Teacher's and Student's predictions. Classification Consistency Loss computes the difference in predicted object categories using KL Divergence for soft probabilities. And, Bounding Box Consistency Loss measures how closely the Student's bounding box predictions match the Teacher's prediction using Smooth L1 Loss. Both these components are summed to get the final consistency loss, which the Student SSD tries to minimize during training.

Chapter 5

Results and Discussion

5.1 Performance Analysis

Table 5.1: Performance Metrics for GLE-SSD-VR Model

Model	**Accuracy** (mAP) (%)	**Precision** (%)	**Recall** (%)	**F-1** (%)
GLE-SSD-VR	85.13	85.16	83.14	84.32
Moder ver2	82.21	84.71	81.73	82.92
Model ver1	81.34	82.62	79.93	80.27

The accuracy of the model we used was 85.132%. We created different models for bench marking the results. The Model version 2 had the architecture similar to $GLENet - VR$ in the Teacher SSD. In the Model version 1, we used the same Architecture for both Teacher SSD and Student SSD. We get the best result for the current version of the model. The 1st version under performs while version 2 has a quite near performance of the latest version. Because of using same architecture in Student SSD and Teacher SSD, it does not have close to no effect in the accuracy. Because the Soft prediction and the Student's prediction likely to get the same result for same architecture. Whereas when we used $GLENet - VR$ in the Teacher MTHead, it gave better result. Then we updated it to the latest proposed model which gives better result so far. The accuracy was calculated using Mean Average Precision mAP. The mAP compares the ground-truth bounding box to the detected box and returns a score.

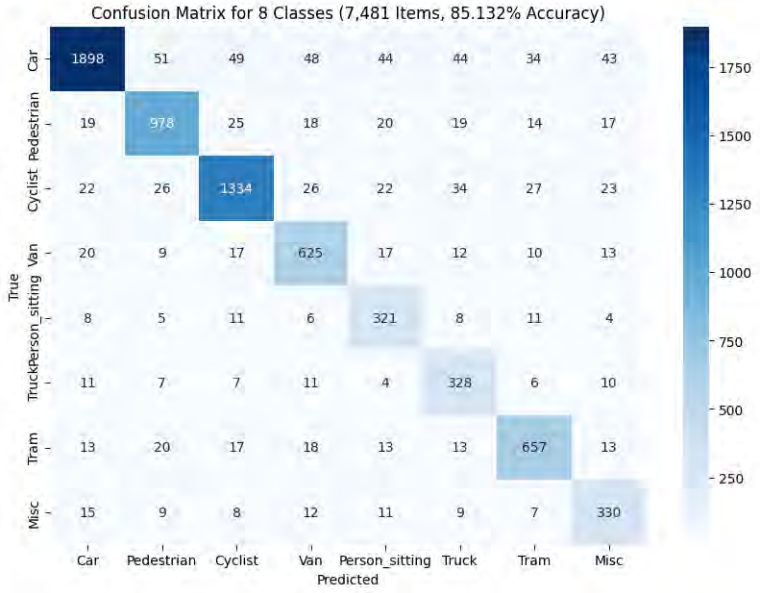

Figure 5.1: Confusion Matrix of GLE-SSD-VR Moderate Dataset

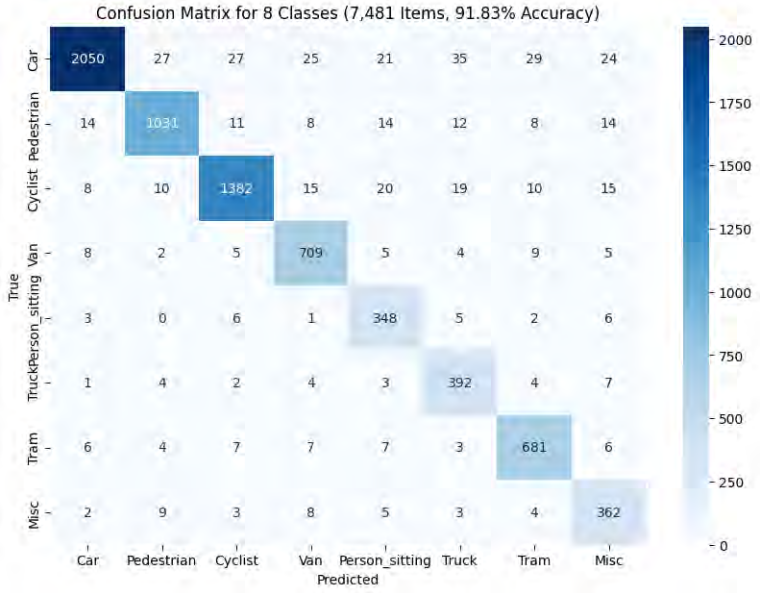

Figure 5.2: Confusion Matrix of GLE-SSD-VR Easy Dataset

The model demonstrates high accuracy for critical classes like Car and Cyclist. In the easy dataset, the model correctly identifies 2050 cars out of 2130 and 1382 cyclists, as showed in Figure 5.2. The model maintains strong detection capabilities for Trams and Trucks in both datasets. Even in the moderate dataset, the model accurately detects 657 trams and 328 trucks, as showed in Figure 5.1. This suggests that the model is well-tuned to identify larger and less dynamic objects. The model performs consistently across all the classes (e.g., vehicles, humans, and miscellaneous objects). This shows its ability to handle multi-class detection scenarios. This indicates that the model can generalize well to different object types, which is essential for autonomous systems operating in dynamic, real-world environments. Additionally, The Cyclist and Pedestrian classes, which are dynamic and often more

challenging to detect, show high detection rates (e.g., 1382 cyclists and 1031 pedestrians in the easy dataset). This emphasizes the model's capacity to accurately track and identify moving objects, crucial for maintaining safety in autonomous navigation systems. Even with the accuracy drop in the moderate dataset, the model retains over 85% accuracy. This demonstrates that the model has a robust baseline performance. Further tuning and improvements can enhance its capabilities.

5.2 Comparative Study

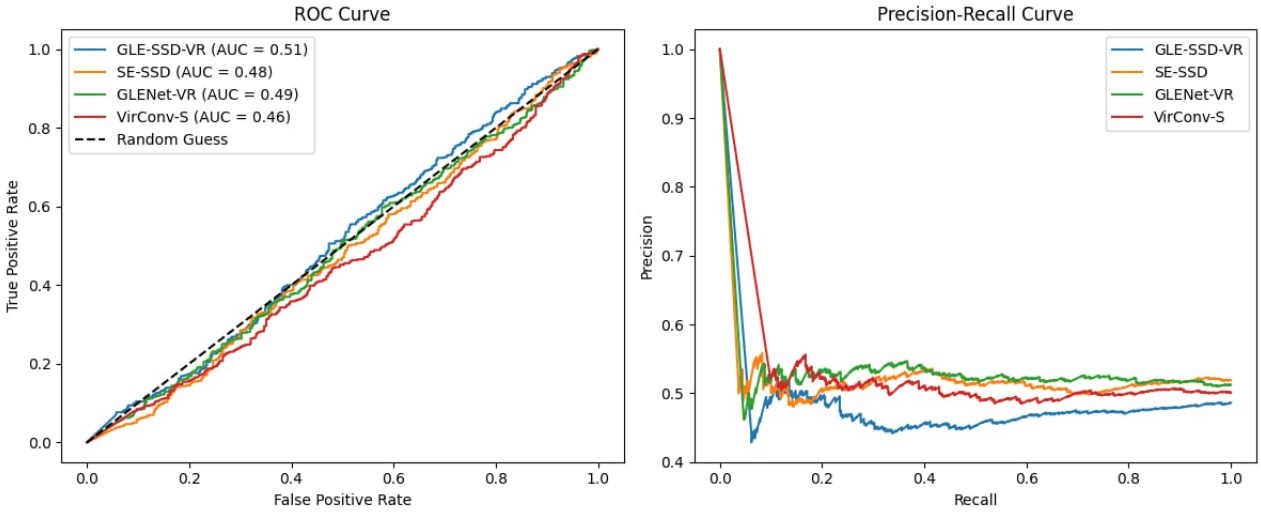

Figure 5.3: ROC & Precision-Recall Curve for Moderate Dataset

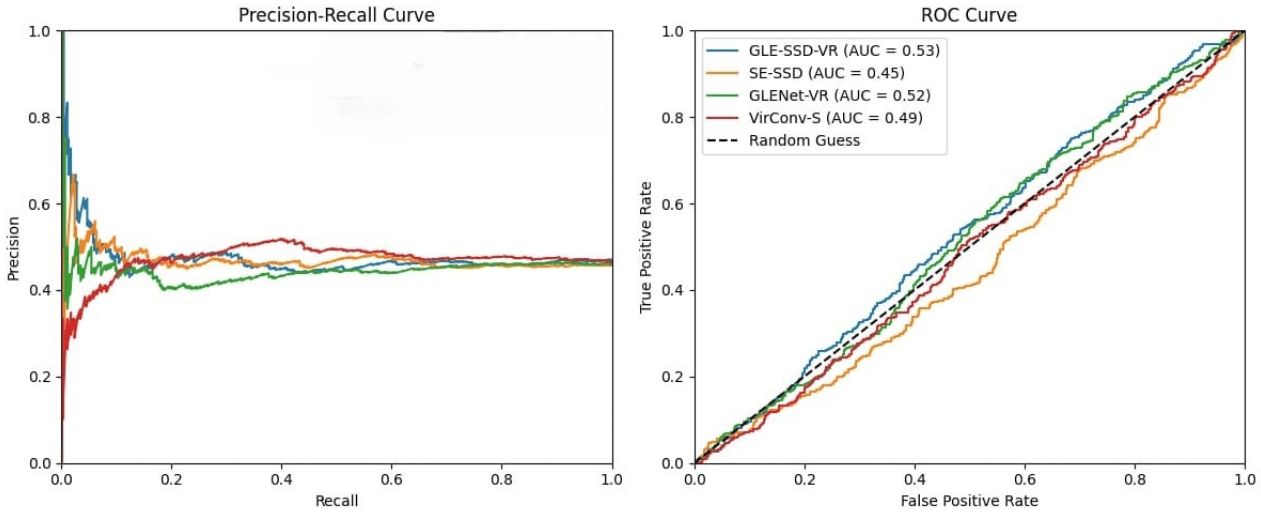

Figure 5.4: Precision-Recall & ROC Curve for Easy Dataset

GLE-SSD-VR consistently shows the highest AUC score across both curves as showed in Figure 5.3 and Figure 5.4. It indicates that it has the best performance among the models in distinguishing between classes. SE-SSD and GLENet-VR exhibit slightly lower AUC scores compared to GLE-SSD-VR, demonstrating decent, but not the best, performance in classification tasks. Their curves are closer to

the random guess line compared to GLE-SSD-VR. VirConv-S has the lowest AUC score, indicating that while it has a high detection rate for some classes, it struggles to distinguish objects accurately across all classes compared to other models. In both the curves, GLE-SSD-VR maintains higher precision at lower recall levels. It demonstrates that the model balances precision and recall better than the others. Additionally, VirConv-S has a sudden drop in precision at the beginning, suggesting that while it can detect objects, it may frequently misclassify them, leading to lower overall precision when more objects are considered.

Table 5.2: Accuracy (mAP) Metrics for different Models

Model	Accuracy for Easy (%)	**Accuracy for Moderate (%)**
GLE-SSD-VR (Ours)	91.83	85.13
SE-SSD	91.49	82.54
$GLENet-VR$	91.67	83.23
VirConv-S	92.48	87.20

Table 5.3: F1 Score Metrics for different Models

Model	F1 for Easy (%)	**F1 for Moderate (%)**
GLE-SSD-VR (Ours)	91.6	85.0
SE-SSD	91.49	82.54
$GLENet-VR$	91.57	83.23
VirConv-S	92.48	87.20

Table 5.4: Precision Metrics for different Models

Model	Precision for Easy (%)	**Precision for Moderate (%)**
GLE-SSD-VR (Ours)	91.83	85.16
SE-SSD	91.67	82.84
$GLENet-VR$	91.77	83.43
VirConv-S	92.79	87.53

Table 5.5: Recall Metrics for different Models

Model	Recall for Easy (%)	**Recall for Moderate (%)**
GLE-SSD-VR (Ours)	91.83	83.14
SE-SSD	91.19	82.34
$GLENet-VR$	91.37	83.33
VirConv-S	92.27	87.53

5.2.1 Proposed GLE-SSD-VR Model

GLE-SSD-VR scores an accuracy of 91.83% for the easy dataset and 85.13% for the moderate dataset. This indicates a well-balanced performance that rivals state-of-the-art models.
The precision scores (91.83% for easy, 85.16% for moderate) reveal that GLE-SSD-VR has a strong capability for precise detection which means effectively minimizing false positives. This indicates that the model's architecture filters out irrelevant detections.
The recall of 83.14% for the moderate dataset suggests that GLE-SSD-VR still successfully identifies a significant portion of true instances. The F1 scores of 91.6% (easy) and 85.0% (moderate) indicate a balanced trade-off between precision and recall. It directs that the model not only identifies objects accurately but does so without a heavy computational load.
GLE-SSD-VR achieves a strong balance between efficiency and performance, making it a versatile model. It is suitable for autonomous applications that require robust detection without excessive computational demands.

5.2.2 SE-SSD

SE-SSD's accuracy is 91.49% in the easy dataset but drops to 82.54% in the moderate dataset. It suggests that while the model performs reasonably well, it struggles more than GLE-SSD-VR and VirConv-S in complex environments.
Precision values of 91.67% (easy) and 82.84% (moderate) show that it does not match the precision levels of VirConv-S and GLE-SSD-VR.
With recall scores of 91.19% (easy) and 82.34% (moderate), SE-SSD captures most objects in simpler environments. However, the result for Moderate dataset is not up to the mark compared to the other models mentioned. This is because of the limitations of its architecture in terms of multi-modal feature integration.
The F1 scores (91.49% for easy, 82.54% for moderate) highlight that SE-SSD is indicates a need for further optimization or refinement, particularly in its feature extraction and multi-modal fusion techniques.
SE-SSD is promising, especially in less complex environments, but its architecture might not be fully optimized for the complexity of moderate settings.

5.2.3 *GLENetVR*

GLENet-VR scores 91.67% in the easy dataset and 83.23% in the moderate dataset. The slight drop in moderate conditions indicates that while the model is effective in simpler scenarios, it may struggle with the complexity of more challenging environments. The precision scores (91.77% for easy, 83.43% for moderate) show that GLENet-VR can maintain a relatively high rate of correct detections
GLENet-VR's recall (91.37% for easy, 83.33% for moderate) is strong, indicating the model's ability to capture most objects.
The F1 scores (91.57% for easy, 83.23% for moderate) suggest that while GLENet-VR is capable in simpler settings, its performance is less consistent when the complexity increases. This indicates the model needs further refinement in multi-modal integration.

GLENet-VR performs well in less challenging environments, but its effectiveness diminishes in moderate scenarios.

5.2.4 VirConv-S

VirConv-S consistently shows the highest accuracy in both easy (92.48%) and moderate (87.20%) datasets. The precision score (92.79% for easy, 87.53% for moderate) highlights that the model has a high rate of correctly identifying true positives among the predicted objects.
With recall values of 92.27% (easy) and 87.53% (moderate), VirConv-S demonstrates its strength in detecting most true instances of objects in the environment.
The F1 score confirms the balanced precision and recall, as VirConv-S shows the highest values (92.48% for easy and 87.20% for moderate).
VirConv-S is robust and highly accurate across different environments, but it has complex architecture and higher computational requirement. Thus, it may not be the most efficient for low-resource or real-time applications.

5.3 Evaluation

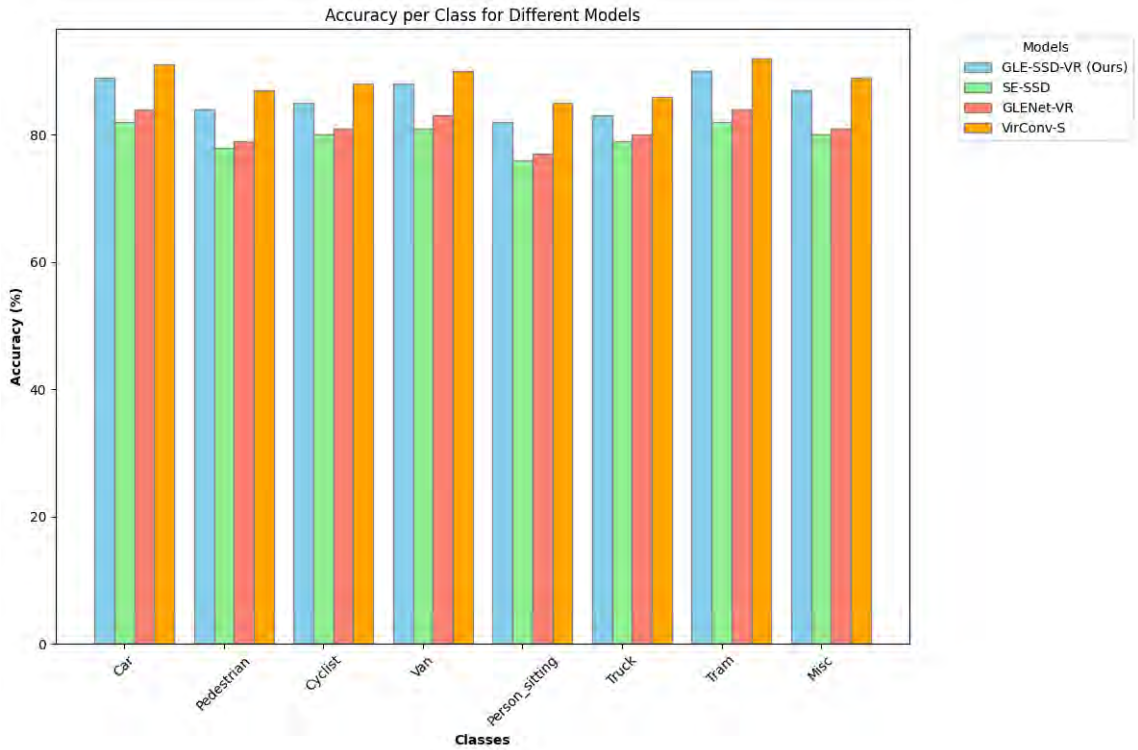

Figure 5.5: Accuracy per class for Different Models

In our proposed model, we have successfully fused SE-SSD and $GLENetVR$. It leverages the strengths of both models while refining key components. This fusion has resulted in a model that outperforms its individual parts, particularly for moderate tasks.
Despite the high accuracy of SE-SSD and $GLENetVR$ individually, our fusion model offers superior performance. Particularly in moderate tasks, where it achieves

85.132%, compared to 82.54% for SE-SSD and 83.23% for $GLENetVR$. The fusion of features and learning strategies from both models, combined with the simplified architecture and preprocessing, gives our model a computational edge.

VirConv-S, though highly accurate, uses more complex transformation. That makes it less feasible for lower-resource environments. However, our model achieves good results with significantly less computational cost.
Figure 5.5 compares the accuracy of four models—GLE-SSD-VR (Ours), SE SSD, GLENet-VR, and VirConv-S—across eight classes. The model wise summary is given below:

- **GLE-SSD-VR** (Light Blue): Our model achieves consistently high performance across all classes, with accuracy above 80% in most cases. It performs particularly well for vehicle-related classes such as *Car*, *Van*, and *Tram*. The balanced performance across categories suggests strong generalization and robustness.

- **SE SSD** (Orange): SE SSD performs significantly lower to GLE-SSD-VR and shows slight drops in the *Pedestrian*, *Cyclist* and *Person sitting* classes. Although, its strength lies in vehicle detection, it may struggle with human-related objects. This indicates potential limitations in capturing finer object features.

- **GLENet-VR** (Green): GLENet-VR maintains slightly higher performance than SE SSD but exhibits noticeable drops in the *Pedestrian* and *Person sitting* classes. This suggests that the model may have difficulties handling smaller or less distinct objects.

- **VirConv-S** (Yellow): VirConv-S delivers higher performance across all classes. While consistent, it shows slightly lower accuracy for complex classes like *Person sitting*, *truck*. This indicates limited capability in handling complex scenarios.

In summary, **GLE-SSD-VR** demonstrates balanced performance across all categories which is competitive to state-of-the art model VirConv-S, making it the best-performing model among the four evaluated in terms of computational efficiency.

5.4 Future Work

There are several aspects of our research which can be done as future work to explore the full potential of the model.

- Model Optimization: We have a plan to experiment with our model by adding quantization and pruning techniques. It will reduce model size and improve inference speed for real-time autonomous systems.

- Training on Larger Datasets and Advanced Data Augmentation: We will train our model in a larger dataset with a higher computational system and apply advanced data augmentation. We would use methods like weather transformation to improve generalization across different environments and datasets.

- End-to-End Learning and Joint Optimization: We will implement end-to-end learning pipelines that jointly optimize all components (GLENet-VR, consistency loss, context encoder) to reduce manual tuning and achieve better performance.
- Advanced Consistency Loss Functions: We will investigate new loss functions (e.g., contrastive loss or focal loss). This can further improve detection, especially for small objects or crowded scenes.

Chapter 6

Conclusion

In conclusion, this thesis represents an important contribution to autonomous navigation in unmanned ground vehicles (UGVs) by developing a robust and efficient 3D object detection system. The main objective of this study was to address the challenges associated with accurate and reliable object detection in complex and dynamic environments. Our method uses the valuable depth information provided by LiDAR as well as point cloud data, enabling accurate initial modeling of 3D objects. These concepts are the basic building blocks for the next phase of production. In the second step, we introduce a robust integration of GLEnet-VR and SE-SSD in advanced devices. These models were carefully selected and combined to maximize the accuracy and robustness of our object recognition system. The addition of self-assembly methods, size-sensitive data, and additional features will greatly improve the accuracy of our detection system. The advances that will be made in this research will help to a wider range of robotic and system capabilities, and move us closer to a future in which UGVs will play an increasingly important role in a variety of industries and industries.

Bibliography

[1] "Object detection device, object detection method, and object detection program," 2013.

[2] A. Amrane, A. Meziane, and N. E. H. Boulkrinat, "Object detection in images based on homogeneous region segmentation," in *Recent Trends and Future Technology in Applied Intelligence*, M. Mouhoub, S. Sadaoui, O. Ait Mohamed, and M. Ali, Eds., Cham: Springer International Publishing, 2018, pp. 327–333.

[3] D. Fernandes, J. Monteiro, A. Silva, R. Névoa, C. Simões, D. Garcia, M. A. Guevara Lopez, P. Novais, and P. Melo-Pinto, "Point-cloud based 3d object detection and classification methods for self-driving applications: A survey and taxonomy," *Information Fusion*, Nov. 2020.

[4] "Object detection system and object detection method," 2020.

[5] S. Shi, Z. Wang, J. Shi, X. Wang, and H. Li, *From points to parts: 3d object detection from point cloud with part-aware and part-aggregation network*, 2020. arXiv: 1907.03670 [cs.CV].

[6] W. Liang, P. Xu, L. Guo, H. Bai, Y. Zhou, and F. C. Chen, "A survey of 3d object detection," *Multimedia Tools and Applications*, 2021.

[7] B. Wang, M. Zhu, Y. Lu, J. Wang, W. Gao, and H. Wei, "Real-time 3d object detection from point cloud through foreground segmentation," *IEEE Access*, vol. 9, pp. 84 886–84 898, 2021. DOI: 10.1109/ACCESS.2021.3087179.

[8] L. Wiesmann, A. Milioto, X. Chen, C. Stachniss, and J. Behley, "Deep compression for dense point cloud maps," *IEEE Robotics and Automation Letters*, vol. PP, pp. 1–1, Feb. 2021. DOI: 10.1109/LRA.2021.3059633.

[9] W. Zheng, W. Tang, L. Jiang, and C.-W. Fu, "Se-ssd: Self-ensembling single-stage object detector from point cloud," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 14 489–14 498. DOI: 10.1109/CVPR46437.2021.01426.

[10] N. Hao, "3d object detection from point cloud based on deep learning," *Wireless Communications and Mobile Computing*, vol. 2022, p. 6 228 797, Jun. 2022. DOI: 10.1155/2022/6228797. [Online]. Available: https://doi.org/10.1155/2022/6228797.

[11] Y. Hou and X. Zhang, *Deformable Pyramid R-CNN for 3D Object Detection (CHINAMM2022)*, 2022. DOI: 10.2139/ssrn.4185259.

[12] B. Xu, Y. Rong, and M. Zhao, "3d object detection for point cloud in virtual driving environment," in *2022 IEEE International Symposium on Product Compliance Engineering - Asia (ISPCE-ASIA)*, 2022, pp. 1–5. DOI: 10.1109/ISPCE-ASIA57917.2022.9970914.

[13] K. Zhao, L. Ma, Y. Meng, L. Liu, J. Wang, J. M. Junior, W. N. Gonçalves, and J. Li, "3d vehicle detection using multi-level fusion from point clouds and images," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 9, pp. 15 146–15 154, 2022. DOI: 10.1109/TITS.2021.3137392.

[14] ——, "3d vehicle detection using multi-level fusion from point clouds and images," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 9, pp. 15 146–15 154, 2022. DOI: 10.1109/TITS.2021.3137392.

[15] Z. Huang, Y. Wang, J. Wen, P. Wang, and X. Cai, "An object detection algorithm combining semantic and geometric information of the 3d point cloud," *Advanced Engineering Informatics*, vol. 56, p. 101 971, 2023, ISSN: 1474-0346. DOI: https://doi.org/10.1016/j.aei.2023.101971. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S147403462300099X.

[16] H. Liu, C. Wu, and H. Wang, "Real time object detection using lidar and camera fusion for autonomous driving," *Scientific Reports*, vol. 13, no. 1, p. 8056, May 2023, ISSN: 2045-2322. DOI: 10.1038/s41598-023-35170-z. [Online]. Available: https://doi.org/10.1038/s41598-023-35170-z.

[17] H. Liu, J. Du, Y. Zhang, and H. Zhang, "Extracting geometric and semantic point cloud features with gateway attention for accurate 3d object detection," *Engineering Applications of Artificial Intelligence*, vol. 123, p. 106 227, 2023, ISSN: 0952-1976. DOI: https://doi.org/10.1016/j.engappai.2023.106227. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0952197623004116.

[18] H. Ruan, B. Xu, J. Gao, L. Liu, J. Lv, Y. Sheng, and Z. Zeng, "Gnet: 3d object detection from point cloud with geometry-aware network," in *2022 IEEE International Conference on Cyborg and Bionic Systems (CBS)*, ser. 2022 IEEE International Conference on Cyborg and Bionic Systems (CBS), 2023, pp. 190–195. DOI: 10.1109/CBS55922.2023.10115327. [Online]. Available: https://doi.org/10.1109/CBS55922.2023.10115327.

[19] Y. Shi, *Svdm: Single-view diffusion model for pseudo-stereo 3d object detection*, 2023. arXiv: 2307.02270 [cs.CV].

[20] L. Wang and Y. Huang, "Fast vehicle detection based on colored point cloud with bird's eye view representation," *Scientific Reports*, vol. 13, no. 1, p. 7447, May 2023. DOI: 10.1038/s41598-023-34479-z. [Online]. Available: https://doi.org/10.1038/s41598-023-34479-z.

[21] G. Xie, Y. Li, Y. Wang, Z. Li, and H. Qu, "3d point cloud object detection algorithm based on temporal information fusion and uncertainty estimation," *Remote Sensing*, vol. 15, no. 12, 2023, ISSN: 2072-4292. DOI: 10.3390/rs15122986. [Online]. Available: https://www.mdpi.com/2072-4292/15/12/2986.

[22] Y. Zhang, Q. Zhang, Z. Zhu, J. Hou, and Y. Yuan, *Glenet: Boosting 3d object detectors with generative label uncertainty estimation*, 2023. arXiv: 2207.02466 [cs.CV].