3D Object Detection with LiDAR

A Summer Internship Report Submitted in the partial fulfillment of the requirements for the award of the degree of

Bachelor of Technology

in

Computer Science and Engineering
-Artificial Intelligence and Machine Learning

Submitted by

G Dharaneesh	22071A6683
Chandradithya Janaswami	22071A6688
Siddharth Verma	22071A66C1
Anurag Sahu	22071A66D3

Under the esteemed guidance of

Mr. K Kishan Babu
Assistant Professor
CSE-AIML & IoT
VNRVJIET

DEPARTMENT OF
COMPUTER SCIENCE AND ENGINEERING-
ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING & INTERNET OF THINGS

VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY

An Autonomous Institute, NAAC Accredited with 'A++' Grade
NBA Accredited for CE, EEE, ME, ECE, CSE, EIE, IT B. Tech Courses
Approved by AICTE, New Delhi, Affiliated to JNTUH
Recognized as "College with Potential for Excellence" by UGC
ISO 9001:2015 Certified, QS I GUAGE Diamond Rated

Vignana Jyothi Nagar, Pragathi Nagar, Nizampet (S.O), Hyderabad – 500 090, TS, India

May-2025

3D Object Detection with LiDAR

A Summer Internship Report Submitted in the partial fulfillment of the requirements for the award of the degree of

Bachelor of Technology
in
Computer Science and Engineering-
Artificial Intelligence and Machine Learning

Submitted by

G Dharaneesh	22071A6683
Chandradithya Janaswami	22071A6688
Siddharth Verma	22071A66C1
Anurag Sahu	22071A66D3

Under the esteemed guidance of

Mr. K Kishan Babu
Assistant Professor
CSE-AIML & IoT
VNRVJIET

COMPUTER SCIENCE AND ENGINEERING-
ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING & INTERNET OF THINGS

VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY
An Autonomous Institute, NAAC Accredited with 'A++' Grade
NBA Accredited for CE, EEE, ME, ECE, CSE, EIE, IT B. Tech Courses
Approved by AICTE, New Delhi, Affiliated to JNTUH
Recognized as "College with Potential for Excellence" by UGC
ISO 9001:2015 Certified, QS I GUAGE Diamond Rated

Vignana Jyothi Nagar, Pragathi Nagar, Nizampet (S.O), Hyderabad – 500 090, TS, India

May 2025

VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY

An Autonomous Institute, NAAC Accredited with 'A++' Grade
NBA Accredited for CE, EEE, ME, ECE, CSE, EIE, IT B. Tech Courses
Approved by AICTE, New Delhi,Affiliated to JNTUH
Recognized as "College with Potential for Excellence" by UGC
ISO 9001:2015 Certified, QS I GUAGE Diamond Rated

Vignana Jyothi Nagar, Pragathi Nagar, Nizampet (S.O), Hyderabad – 500 090, TS, India

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING & INTERNET OF THINGS

CERTIFICATE

This is to certify that the project work entitled **"3D Object Detection with LiDAR"** is being submitted by G Dharaneesh (22071A6683), Chandradithya Janaswami (22071A6688), Siddharth Verma (22071A66C1), Anurag Sahu (22071A66D3) in partial fulfillment for the award of Degree of Bachelor of Technology in CSE-Artificial Intelligence and Machine Learning to the Jawaharlal Nehru Technological University, Hyderabad during the academic year **2024-25** is a record of bonafide work carried out under our guidance and supervision. The results embodied in this report have not been submitted by the students to any other University or Institution for the award of any degree or diploma.

Supervisor

Mr. K Kishan Babu
Assistant Professor
CSE -AIML & IoT
VNRVJIET

Head of the Department

Dr. Sagar Yeruva
Associate Professor
CSE-AIML & IoT
VNRVJIET

Signature of the External Examiner with date

VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY

An Autonomous Institute, NAAC Accredited with 'A++' Grade
NBA Accredited for CE, EEE, ME, ECE, CSE, EIE, IT B. Tech Courses
Approved by AICTE, New Delhi, Affiliated to JNTUH
Recognized as "College with Potential for Excellence" by UGC
ISO 9001:2015 Certified, QS I GUAGE Diamond Rated

Vignana Jyothi Nagar, Pragathi Nagar, Nizampet (S.O), Hyderabad – 500 090, TS, India

DEPARTMENT OF
COMPUTER SCIENCE AND ENGINEERING
ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING & INTERNET OF THINGS

DECLARATION

We hereby declare that the project entitled **"3D Object Detection with LiDAR"** submitted to VNR Vignana Jyothi Institute of Engineering and Technology in partial fulfillment of the requirement for the award of Bachelor of Technology in CSE - Artificial Intelligence and Machine Learning is a bonafide report of the work carried out by us under the guidance and supervision of Mr. Kishan Babu, Assistant Professor, Department of CSE -Artificial Intelligence and Machine Learning & Internet of Thing, VNRVJIET.

To the best of my knowledge, this has not been submitted in any form to any university or institution for the Award of any degree or diploma.

G Dharaneesh	Chandradithya Janaswami	Siddharth Verma	Anurag Sahu
22071A6683	22071A6688	22071A66C1	22071A66D3

Place: Hyderabad

Date: 23-05-2025

ACKNOWLEDGMENT

Behind every achievement lies an unfathomable sea of gratitude to those who activated it, without which it would never have come into existence. To them, we lay the words of gratitude imprinted within us.

We are indebted to our venerable principal **Dr. C. D. Naidu** and our Head of the Department **Dr. Sagar Yeruva,** for the support and encouragement given by them led us to complete this Summer Internship Project.

We express our thanks to our project guide **Mr. K Kishan Babu** and the Project Coordinator **Mr.Shaik Mabasha** for having provided us with a lot of facilities to undertake the project work and guide us to complete the Summer Internship project.

We express our sincere thanks to our faculty of the Department of CSE-Artificial Intelligence and Machine Learning & Internet of Things and the remaining members of our college **VNR VIGNANA JYOTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY** who extended their valuable support in helping us to complete the project in time.

1. G Dharaneesh – 22071A6683 _____
2. Chandradithya Janaswami – 22071A6688 _____
3. Siddharth Verma – 22071A66C1 _____
4. Anurag Sahu – 22071A66D3 _____

Abstract

In recent times there has been an advancement in technologies such as autonomous vehicles, robotics and etc, where the need to perceive the environment becomes of paramount importance. The typical 2D vision models fail to provide the depth and spatial awareness needed for such applications. As a result there has been a rise in the usage of LiDAR data to get a 3D view of the scene.

In this project we implement a LiDAR based 3D object detection system using opensource tools. we used the KITTI dataset which comprises of labelled LiDAR point cloud data. we chose to implement a point pillar model for the computational efficiency it offers making it feasible to deploy in real-time scenarios. The pipeline consists of extracting the point clouds and reconstructing them into 2D pseudo images followed by convolution operations to extract features and real time inference to detect objects and classify them.

Table of Contents

List of Figures

CHAPTER 1: INTRODUCTION

1.1 Purpose of the Project

The purpose of the project is to develop a system that can swiftly and effortlessly detect and identify different objects from the data that is captured using a LiDAR, which is usually in the form of a 3d image. The practical use of this project extends to various different applications such as self-driving cars , robotics and drones, Security and Surveillance, etc.

1.2 Existing methodologies and Disadvantages

Various methodologies exist for detecting objects from lidar data each with distinct advantages and disadvantages. Point based methods which work on the LiDAR point clouds directly hence preserving the various geometric details of an object,but they come with a high computational cost and are very difficult to scale to large environments. Voxel-Based methods convert the point clouds into voxel grids,i.e into 3d pixels and apply 3d convolutions on them making it possible to leverage the power of deep learning,the problem with this approach is that it consumes a lot of memory for high resolution images making it less feasible to implement in end user devices. Birds-eye-view based approaches work by translating the point clouds into a two dimensional top-down view image and applying convolutions to extract features,however these approaches lead to a loss in vertical height and depth information. Multi modal Fusion methods combine the LiDAR images with their corresponding RGB images to detect objects, these methods would require a complex implementation system and a high precision calibration between the LiDAR and the RGB camera. Finally, the choice of a methodology is determined by various factors such as budget, accuracy requirements, scalability, and user preferences, with possible benefits in combining techniques or tailoring solutions to individual needs.

1.3 Proposed System

The model's architecture is specially designed to work quickly and effectively in real time environments.The model is built using OpenPCDet, an open source framework built on Pytorch specifically for the task of 3D object detection. We chose the point pillars model for this task.In the first stage the system inputs the raw 3D point cloud data from the LiDAR and preprocessing is done on the data to remove noise and outliers.Then the points are arranged into vertical columns or 'pillars'. A PointNet-style mini-network is then used to extract meaningful features from the pillars, these features are then used to construct a 2D pseudo image.This image is processed using a convolution neural network(CNN) which extracts the spatial features and as its a standard 2D image the processing is quick and effective. The output of the CNN is fed into a detection head which predicts 3D bounding boxes,class labels, and confidence scores. In the next stage , as a part of post processing , Non-Maximum Suppression (NMS) is applied to handle overlapping bounding boxes.The final detections are then visualized using Bird's Eye View (BEV).

1.4 Objective

The core objective of the project is to develop a robust, reliable,light-weight and easy to deploy model that swiftly and effectively detects objects from 3D LiDAR data. The process begins with utilizing benchmark data-sets such as KITTI which includes many different scenes and environments making the model effective when deployed in real time. the model learns to reconstruct the 3D point cloud data in 2D pseudo images to utilize the faster processing capabilities of the 2D CNNs. The project is aimed towards enabling various applications such as robotics and self-driving vehicles to have a better understanding of the surroundings and navigate through various scenarios effectively.

1.5 Thesis Organization

Chapter 1: This chapter presents the basic information and introduction and the necessary technical knowledge to implement.

Chapter 2: This chapter presents the literature survey and research needed to implement the project.

Chapter 3: This chapter presents the challenges and issues that occurred during the implementation of the project.

Chapter 4: This chapter presents the software requirement analysis.

Chapter 5: This chapter presents the software design with UML diagrams.

Chapter 6: This chapter presents the methodology and approach.

Chapter 7: This chapter presents the implementation of the methodology.

Chapter 8: This chapter presents the experimental results of the model.

Chapter 9: This chapter presents the conclusion.

Chapter 10: Future Scope of the Project

Chapter 11: References collected for the Project.

Chapter 12: Source code of the Project

CHAPTER 2: LITERATURE SURVEY

In this project, we focus on three widely recognized benchmark datasets for 3D object detection: KITTI[4], Waymo[6], and nuSCENES[5]. Our work delves into the workings of current State-of-the-Art (SOTA) models, analyzing their methodologies, performance, and underlying technologies. One particularly intriguing insight we uncovered is that native models, when evaluated on these datasets, often do not perform optimally in isolation. Their performance significantly improves when integrated with domain-specific solutions that address challenges such as domain gaps, label uncertainty, and object occlusion. These factors play a critical role in bridging the gap between model generalization and real-world application. Through this research, we aim to highlight the importance of adapting SOTA models to the unique complexities posed by each dataset and domain-specific problem, thus offering a deeper understanding of the advancements and potential limitations in current 3D object detection approaches.

2.1. GLENet-VR

Variational Auto-Encoders are used. This is an extension to existing nets which attempts to quantify label uncertainty derived from some simple heuristics, i.e it doesn't take the given GT labels to be perfectly correct. Thus, this is a probablistic detector. This introduces a latent variable to capture the distribution over potentially plausible bounding boxes of point cloud objects.

There a 4 networks in the working of the GLENet:

• Contect encoder: takes in a Cloud Point C and transforms it into a high dimensional vector space fc after the parameters are learnt.

• Prediction Network: fc fed into a prediction network composed of MLPs to regress the bounding box distribution p(X —z, C), i.e., the localization, dimension, and orientation of the bounding box.

• Prior Network: Composed of a PointNet Base Model and additional MLP layers

• Recognition Network: to generate point cloud embeddings, which are concatenated with ground-truth bounding box information and fed into the subsequent MLP layers to learn q (z —X, C).

The proposed GLENet-VR which is integrated on VoxelNet RCNN[2], surpasses all published single-modal detection methods by a large margin and ranks the best among all published LiDAR-based approaches on the highly competitive KITTI 3D detection benchmark.

2.2 BTC Net

BtcNet [BEHIND THE CURTAIN NET] [15] is the first 3D object detector that targets the object shapes affected by occlusion. With the knowledge of shape priors, BtcDet estimates the occupancy of complete object shapes in the regions affected by occlusion and signal miss. BtcDet improves 3D object detection by identifying regions of occlusion and signal miss in LiDAR point clouds, then using a Shape Occupancy Network to estimate object shape occupancy. These estimates are integrated with a backbone network to generate and refine 3D bounding box proposals. BtcDet combines occupancy prediction and bounding box supervision for more accurate detection.

The Shape Occupancy Network, proposed by Xu et. al, is a novel approach as there is no existing base model. This unique method enhances object detection by leveraging partial information, which significantly improves detection accuracy in scenarios where traditional models struggle with occlusions or incomplete data.

BtcDet operates through a multi-step pipeline to improve 3D object detection in LiDAR-based point clouds, especially in occluded or partially visible scenarios:

1. Region Identification:

• First, BtcDet identifies regions of occlusion (ROC) and regions of signal miss (RSM). These regions are critical because occlusions and missed signals hinder accurate object detection.

2. Shape Occupancy Estimation:

• Using a Shape Occupancy Network (Ω), BtcDet estimates the probability of object shape occupancy (P(OS)), which predicts whether certain regions in the point cloud are likely to be occupied by objects (shown as orange voxels where P(OS) > 0.3). This step helps in handling occlusions by inferring missing parts of objects.

3. Feature Extraction and Region Proposals:

• A backbone network (Ψ) extracts 3D features from the LiDAR point cloud. These features are concatenated with the occupancy probability map (P(OS)), which provides additional context about object shape.

• The Region Proposal Network (RPN) generates potential 3D bounding box proposals from the extracted features.

4. Proposal Refinement:

• For each proposal, local geometric features (fgeo) are pooled from nearby grids and combined with the occupancy information and multi-scale features from Ψ.

• The final bounding box prediction is refined using these pooled features, resulting in a more accurate 3D object detection outcome.

5. Optimization:

• The model is optimized using two objectives:

– Shape occupancy prediction (P(OS)): Estimating the likelihood that a given region is occupied by an object.

– Bounding box prediction (X, D): Estimating the center and dimensions of the object based on the point cloud and occupancy information.

2.3. 3D Dual Fusion

Transformer Fusion Based Model. It fuses the features of the camera-view and 3D voxel-view domain and models their interactions through deformable attention [16].

The architecture involves designing a Dual-Fusion Transformer architecture[17] consisting of 3D local self-attention and dual-query deformable attention. This extensively employs dual query deformable attention mechanism on which we need to read more.

TransFusion[18] and CenterPoint[19] Model's Detection head has been used to build the Model Of 3D Dual Fusion, while the attention mechanism is formulated by the authors.

2.4. Semantic Point Generation Networks

This Model focuses on improving 3D object detection in autonomous driving, particularly addressing the domain gap[20] caused by environmental factors like rain, which degrade Li-DAR data quality. A novel method, Semantic Point Generation (SPG), is introduced to enhance the reliability of LiDAR based detectors in unsupervised domain adaptation (UDA)

scenarios. SPG addresses the ”missing point” problem by predicting and generating semantic points in key foreground areas, improving object detection in both source and target domains. Previous methods, such as random down-sampling and style transfer, fail to capture the irregular patterns of missing points or introduce inefficiencies. SPG overcomes these limitations by predicting foreground regions and generating points based on semantic information, significantly boosting detection performance. The model integrates easily with existing detectors like PointPillars[21], PV-RCNN[22] and PVRCNN++[23], enhancing accuracy without adding significant computational overhead. Key strategies like ”Hide and Predict” and ”Semantic Area Expansion” are used to mimic real-world point cloud degradation during training, ensuring robust performance in various conditions.

2.5. PV-RCNN

PV-RCNN introduces a groundbreaking framework aimed at enhancing the accuracy of 3D object detection from point clouds. This approach synergizes the strengths of both voxel based and point-based networks, significantly improving feature extraction and detection performance. The method integrates voxel and point features by merging 3D voxel Convolutional Neural Network [24] with PointNet [21]based set abstraction. This integration enables the model to exploit the efficient learning capabilities of voxel CNNs while harnessing the flexible receptive fields characteristic of pointbased methods.

The method employs a voxel CNN to summarize the 3D scene, generating high-quality proposals. These proposals are then processed through a novel voxel set abstraction module, which distills the data into a compact set of keypoints, thereby enhancing computational efficiency. To further enhance the model’s accuracy, the framework implements a RoI-grid pooling module for each box proposal. This module aggregates features from keypoints to the grid points linked with the proposals, utilizing a keypoint set abstraction layer with multiple radii to capture rich contextual information crucial for accurate object confidence and location estimations.

The framework also focuses on proposal refinement by concentrating on keypoints pertinent to the foreground, thereby enhancing multi-scale feature aggregation. This process is facilitated by a predicted keypoint weighting (PKW) module, which emphasizes the significance of foreground keypoints during the refinement process.

The effectiveness of the PV-RCNN[22] framework is thoroughly validated through extensive experiments on the KITTI[4] and Waymo Open datasets[6]. The results indicate that this method substantially surpasses previous state-of-the-art 3D detection approaches, highlighting its proficiency in leveraging point clouds for object detection.

2.6. PV-RCNN++

This Paper introduces an enhanced framework for 3D object detection, aiming to improve efficiency and accuracy in processing point clouds. Building on the original PVRCNN[22] framework, the authors implement two significant enhancements: sectorized proposal-centric sampling and VectorPool[23] aggregation. By integrating point-based and voxel-based methods, PV-RCNN++ effectively leverages the strengths of both, addressing the computational intensity of point-based methods and the quantization errors inherent in voxel-based approaches.

The framework begins with voxelizing[13] the input point cloud, followed by sectorized proposal-centric sampling to select representative keypoints from voxel centers, ensuring they accurately reflect the underlying data structure. This innovative sampling strategy is approximately five times faster than traditional farthest point sampling, reducing the number of candidate keypoints and enhancing detection performance. Additionally, the VectorPool[23] aggregation technique improves the efficiency of local point feature aggregation, consuming less memory while maintaining performance. Extensive experiments on the Waymo Open Dataset[6] demonstrate that PVRCNN++ achieves state-of-the-art performance in 3D object detection, being about three times faster than its predecessor, PV-RCNN[22], while also improving detection accuracy.

2.7. Voxel-RCNN

This Paper focuses on an overlooked approach. It is considered that Voxel-based representation causes loss of information, and so most high-performance 3D object detectors are point-based. This paper takes a different approach assuming that the precise position of raw points is not required for high performance, and that voxels can provide sufficient performance. In Voxel-RCNN, the raw point cloud is split into voxels, and then run through a 3D backbone network to extract features, then these sparse Voxels are sent to a 2D backbone network and an RPN (region proposal network) to generate the 3D region

proposals. Then we use Voxel RoI pooling, to extract RoI (regions of interest) features, which are then fed into the detect subnet for box refinement. The effectiveness of the VoxelRCNN is validated when tested on KITTI benchmark[4], and Waymo Open Dataset[6]. The Model beats all state-of-the-art models on the Waymo Open dataset on both levels 1 and 2. The model achieves around state-of-the-art efficiency on the KITTI benchmark, Highlighting that Voxel-based representation can give give good performance with less computational complexity, with the help of a little tuning.

2.8. BEVFusion

This Paper focuses on multi-sensor fusion, and what kind of fusion, will give the best result, as LiDAR data is classified as 2.5D and camera data as 2D, it can lead to feature space discrepancies. This approach unifies multi-modal features in a common Bird's Eye View (BEV). Overcoming both Semantic loss when using only LiDAR, and geometric loss when using only a camera. It performs Efficient camera-to-BEV conversion, by performing precomputation and interval reduction, making the transformation faster by around 40X [1]. The models used are, Swin-T as the camera backbone and VoxelNet as the LiDAR backbone. The effectiveness of the BEVFusion is validated when tested on nuScenes dataset[5], and Waymo Open Dataset[6]. The Model gives better results when tested on same data with a standalone LiDAR. The model is also able to give satisfactory results in rainy weather, compared to Standalone LiDAR, where it provides unsatisfactory results.

2.9. EA-LSS

This paper introduces EA-LSS, Edge-Aware Lift-Splat-Shoot, an increment on previous LSS based 3D object detection work. It is noticed that most BEV-oriented methods struggle with depth discontinuities and faulty feature projection, to tackle this EA-LSS uses Edge-Aware Depth Fusion (EADF) and Fine-Grained Depth (FGD) modules to improve depth estimation and enhance BEV quality [25]. In this method, the input camera image undergoes LSS style feature lifting followed by EADF, to take care of the depth jump issues

by preserving spatial edge information. Then the FGD module is used, which improves depth guided supervision by using LiDAR depth signals to fine-tune the BEV representations.

Then these refined BEV features are passed on to the detection head, improving accuracy and precision [25].

The performance of EA-LSS is evaluated on the nuScenes dataset [5], where it is able to achieve state-of-the-art results. The model beats previous depth-supervised works and achieves considerable performance against LiDAR-based detectors. Despite it's state-of-the-art performance it isn't computationally expensive making it suitable for real-time applications.

2.10. TransFusion

This paper introduces TransFusion, a novel transformer based LiDAR-camera fusion framework for 3D object detection. It is noticed that most of the multi-modal detection approaches depend on hard LiDAR image associations [26]. This paper proposes a soft association mechanism, allowing for flexible fusion of features and tackling sensor alignment issues effectively. A general assumption made in this approach is that direct LiDAR-camera correspondences are not necessary for high performance and that adaptive fusion using transformers can give more robust 3D perceptions [18]. The pipeline is that, first the raw LiDAR point cloud is processed through a 3D backbone network to give region proposals, while the camera image is encoded separately. Then a two-stage transformer decoder refines the bounding box predictions by integrating image features into the LiDAR-based object queries. The first decoder generates object proposals purely from LiDAR, while the second decoder selectively incorporates image features, guided by an image-aware query initialization strategy [18]. This allows the model to perform better even with small, occluded objects.

The performance of TransFusion is validated on the nuScenes and Waymo dataset [5][6], where it is able to achieve state-of-the-art performance in 3D object detection. The model beats previous LiDAR-only and multi-modal methods, Showcasing the viability of

transformers for object detection. It performs well in low light and with sensor noise, because of it's transformer-based adaptive fusion. A major drawback is the computational cost of transformers due to it's complex nature [27].

2..11. MSMDFusion

This paper introduces MSMDFusion, a novel multi-scale LiDAR-camera fusion framework for 3D object detection. It is noticed that most of the LiDAR-camera fusion methods either suffer from poor depth alignment or over-simplified BEV projections. This paper proposes a multi-depth unprojection (MDU) module to refine virtual depth points and a Gated Modality-Aware Convolution (GMA-Conv) for fine-grained feature fusion [28]. It is assumed that multi-scale depth-aware fusion can bring significant improvement in 3D perception quality with little computational impact. In MSMDFusion, the pipeline is that, first the raw LiDAR point cloud is processed to extract voxelised features, while the camera image is used to get the depth-aware virtual points through the MDU module. The GMA-Conv module then integrates the multi-modal features allowing for adaptive interaction between depth-aware image features and LiDAR signals. This allows the model to perform better even with distant and occluded objects. The performance of MSMDFusion is validated on the nuScenes dataset [5], where it is able to achieve state-of-the-art performance in LiDAR-camera fusion. The model beats previous fusion methods, demonstrating the viability of multi-scale depth-aware feature integration. The computational cost of this method makes it viable for usage in autonomous driving applications.

2.12. PointRCNN

This paper introduces PointRCNN, a two-stage method to detect objects. The first stage is used for gen- erating 3-D proposals from raw point cloud in a bottom- up manner, and the second stage is used for refining the proposals in canonical coordinates [10]. This paper utilizes the raw point features for precise localization. It is assumed that point-wise instance segmentation and proposal refinement can lead to high accuracy. Compared to previous works which use 2-D images or voxelised representations, this approach segments the point

cloud into background and foreground to generate 3-D proposals. A new loss function is introduced to improve the precision of the bounding boxes.

The performance of PointRCNN is tested on the KITTI 3D object detection benchmark [4], where it is able to achieve stateof- the-art accuracy in point-based methods. The model is able to beat voxel based methods in some cases. The high computational cost and dependency on high-density point cloud doesn't make it a practical choice for real world applications.

2.13. SupFusion

This paper introduces SupFusion, a supervised LiDAR camera fusion strategy for 3D object detection. Majority of LiDAR-camera fusion based approaches perform feature extraction followed by direct fusion. They lack proper supervision in the fusion process, potentially reducing performance. A new feature-level auxiliary supervision strategy is proposed, where a feature assistant model guides the fusion network to generate better multi-modal representations [29]. SupFusion introduces a new data augmentation technique, Polar Sampling which improves sparse LiDAR point clouds by densifying objects based on their direction and rotation. This allows the assistant model to generate better LiDAR features. The effectiveness of SupFusion is validated on the KITTI 3D object detection benchmark [4], where it is able to achieve a 2% improvement in mAP over existing LiDAR-camera fusion models. It is a plug and play module, making it usable directly with many LiDAR-camera 3D Detectors. Due to the model being trained only on the KITTI benchmark, the performance of the model remains uncertain in real-world scenarios.

CHAPTER 3: ISSUES AND CHALLENGES

The implementation of effective 3D objection through LiDAR faces multiple challenges that must be addressed to ensure its successful operation.

One significant challenge is the high computational cost involved with processing 3D data. To allow for real-time inference, high-end GPUs/CPUs are required. Also the accuracy of the model will depend on the data representation method chosen, like Voxels, Point Clouds, Bird's Eye View, the choice here will impact the performance, speed and memory usage.

Moreover, for real-time use cases, the processing has to be fast as we cannot afford high latency. A desirable latency (<30ms) is achievable through hardware acceleration and efficient software.

An additional challenge arises from the sensor fusion compatibility. 3D data is often sourced from a combination of sensors such as LiDARs, cameras, IMUs, etc. This can further complicate the task of data preprocessing and synchronization challenges. Existing sources of 3D data are not annotated, requiring manual annotation which is time consuming.

Furthermore, real world scenarios include distortions and noise created by weather conditions such as rain, smoke, haze can reduce the visibility, effectively reducing the performance of the model. Also, real-world generalization is an issue, as models trained in a specific environment will struggle to adapt in new situations.

Lastly, the problem of power consumption of embedded devices running such high-power models is massive, huge batteries are required for continuous use, leading to more logistical problems. Downscaling of algorithms like network pruning and hardware optimization is required for real-world use.

Addressing these challenges through strategic planning and innovative solutions is crucial for the successful implementation of a 3D object detection system. By overcoming these problems, we can enhance the efficiency of real-time systems like self-driving cars, automated surveillance systems and so on. The listed problems highlight the nature of new age technologies and the downsides that come with them.

CHAPTER 4: SOFTWARE REQUIREMENT ANALYSIS

A requirements analysis is a vital step in assessing the success of a system or a software project. Requirements are two types: mandatory and optional.

- Functional requirements
- External Interface requirements
- Non-functional requirements

4.1 Functional Requirements

4.1.1 Definition

These are the necessities that the end client communicates as fundamental elements that the framework ought to give. As a feature of the agreement, these functionalities should be incorporated into the framework. These are communicated or depicted as contributions to be conveyed to the framework, activity to be directed, and anticipated yield. They are the client's communicated prerequisites that, in contrast to non-utilitarian models, should be visible promptly in the finished item.

4.1.2 Requirements for this project

- Efficient point cloud conversion.
- Real-time object detection.
- Continuously process sensor data.
- Sync and fuse the sensor data efficiently.
- Regular functioning in adverse weather conditions.
- Reliable and accurate detection.
- Save detection results in a log file or database.
- Output bounding boxes with labels.
- Versatility to deploy across various frameworks
- Comprehensive documentation provided.

4.2 Non-Functional Requirements

4.2.1 Definition

Non-functional requirements specify the qualities or constraints that the system must exhibit. These requirements focus on characteristics such as performance, reliability, usability,

scalability, security, and maintainability. Non-functional requirements are typically more subjective and harder to quantify than functional requirements.

4.2.2 Requirements for this project

- Reliable and accurate
- A user-friendly UI
- No leakage of sensor data
- Modular code and architecture for easy debugging
- Should handle noisy/missing data gracefully
- Less processing time for Detection
- Optimized for power consumption

4.3 External Interface Requirements

4.3.1 Definition

Functional requirements include the necessity for external interfaces. They are essential parts of embedded systems. They also explain how your product will work in conjunction with other features. You may require a variety of interfaces, including:

- User
- Hardware
- Software
- Communications

4.3.2 Requirements for this project

The below are the software required for the python app in this project

- **Python:**

  Python is an interpretable high-level programming language. Python provides concise and understandable code. While complex computations and adaptive work processes stand behind AI, Python's simplicity allows designers to compose trustworthy frameworks. Designers can devote all their efforts to addressing an ML issue rather than focusing on the language's particular nuances. It uses an article-based approach to provide customers with a reasonable perspective on the activities' intellectual and practical components.

- **Google Colab:**

Colaboratory, simply called "Colab", is a Google Research product. Colab allows anyone to write and execute inconsistent Python code through the application, and it is especially useful for AI, information investigation, and instruction. Furthermore, Colab is a streamlined Jupyter Notebook administration that demands no setup while providing free access to training components such as GPUs. Jupyter is the open-source project upon which Colab is built. Colab allows you to use and share Jupyter notebooks with others without downloading, setting up, or running anything.

Google Colab provides all the exciting features that any modern IDE does, plus much more. Probably the most interesting components are listed here.

- Intuitive learning activities for AI and brain organizations.
- Compose and run Python 3 code without a neighborhood structure.
- Execute terminal commands from the Notebook.
- Import datasets from outside sources, such as Kaggle.
- Save your Notebooks to Google Drive.
- Import Notebooks from Google Drive.
- Cloud administration, GPUs, and TPUs are all provided for free.
- Coordinate with PyTorch, TensorFlow, and OpenCV.

- **Roboflow:**

Roboflow is a comprehensive platform designed to streamline the management of computer vision datasets. It offers intuitive tools for annotating images with bounding boxes, polygons, and semantic segmentation masks. Additionally, Roboflow simplifies preprocessing tasks by providing resizing, cropping, and normalization capabilities. With built-in data augmentation techniques like flipping, rotating, and adding noise, users can generate diverse and representative datasets for training machine learning models. The platform supports collaboration and version control, enabling seamless teamwork on large-scale projects. Furthermore, Roboflow facilitates model deployment with integration options for popular deployment platforms like TensorFlow Serving and TensorFlow Lite.

Overall, Roboflow empowers developers, researchers, and businesses to efficiently annotate, preprocess, augment, and deploy image datasets for various computer vision applications. Its scalable and performance-driven cloud-based architecture accelerates data processing and model training, ensuring faster iteration cycles and quicker time-to-market for ML projects. By providing a broad set of tools and

features, Roboflow enables users to overcome the challenges associated with managing computer vision datasets and achieve success in their computer vision endeavors.

- **CUDA:**

  CUDA (Compute Unified Device Architecture) serves as an important external interface requirement which helps in leveraging GPU acceleration for our 3D object detection system. It provides the necessary API and runtime environment for properly utilising the parallel processing capabilities of NVIDIA GPUs. CUDA integration allows for processing of complex 3D data within real-time constraints. Therefore, ensuring CUDA compatibility is essential for model deployment, particularly when using heavy frameworks like PyTorch or TensorFlow with GPU support.

And the libraries used for the python app in this project are:

- Keras
- Tensorflow
- Streamlit
- OpenCV
- NumPy
- PyTorch

CHAPTER 5: SOFTWARE DESIGN

5.1 System Architecture

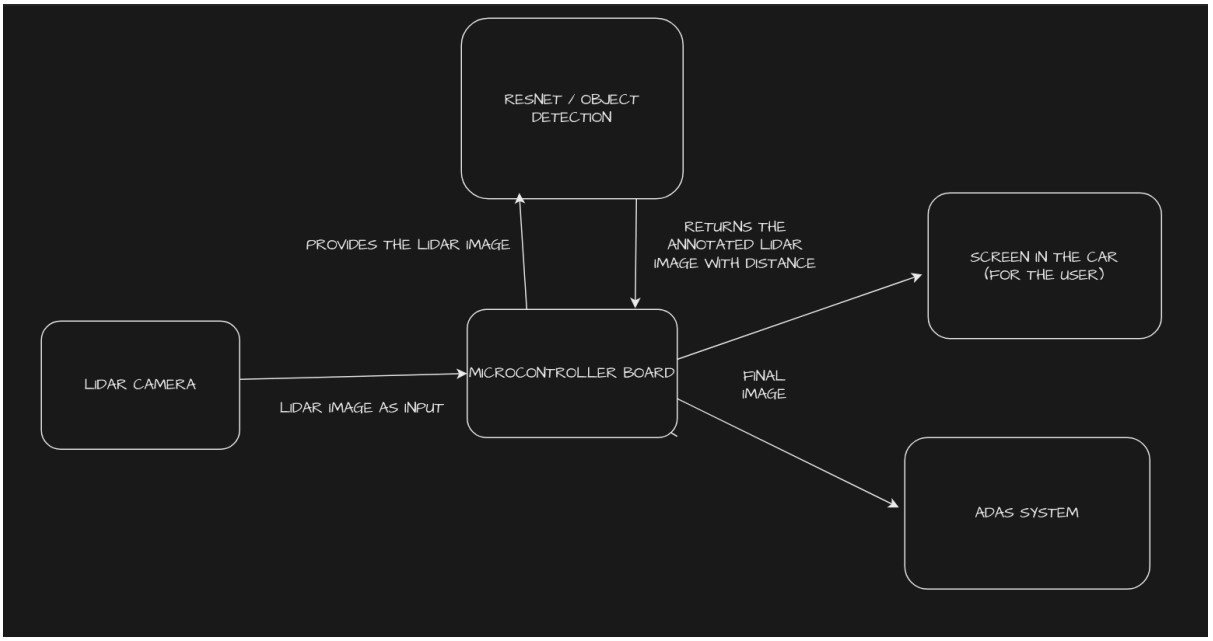

Figure 5-1: System Architecture

5.2 UML Diagrams

Unified Modeling Language (UML) diagrams are graphical representations used to depict a software system's structure, behavior, relationships, and architecture. UML diagrams are popular in software engineering because they provide a standardized syntax for discussing and documenting software system architecture. UML diagrams are classified into structural (class and object diagrams) and behavioral (activity and state machine diagrams).

5.2.1 Class Diagram

A class diagram is a UML diagram depicting a system's structure using classes, characteristics, methods, and their interactions. It gives a clear overview of the system's object-oriented architecture and is commonly used during the software development process to help developers understand the structure of the system they are building.

The following is the class diagram of the system:

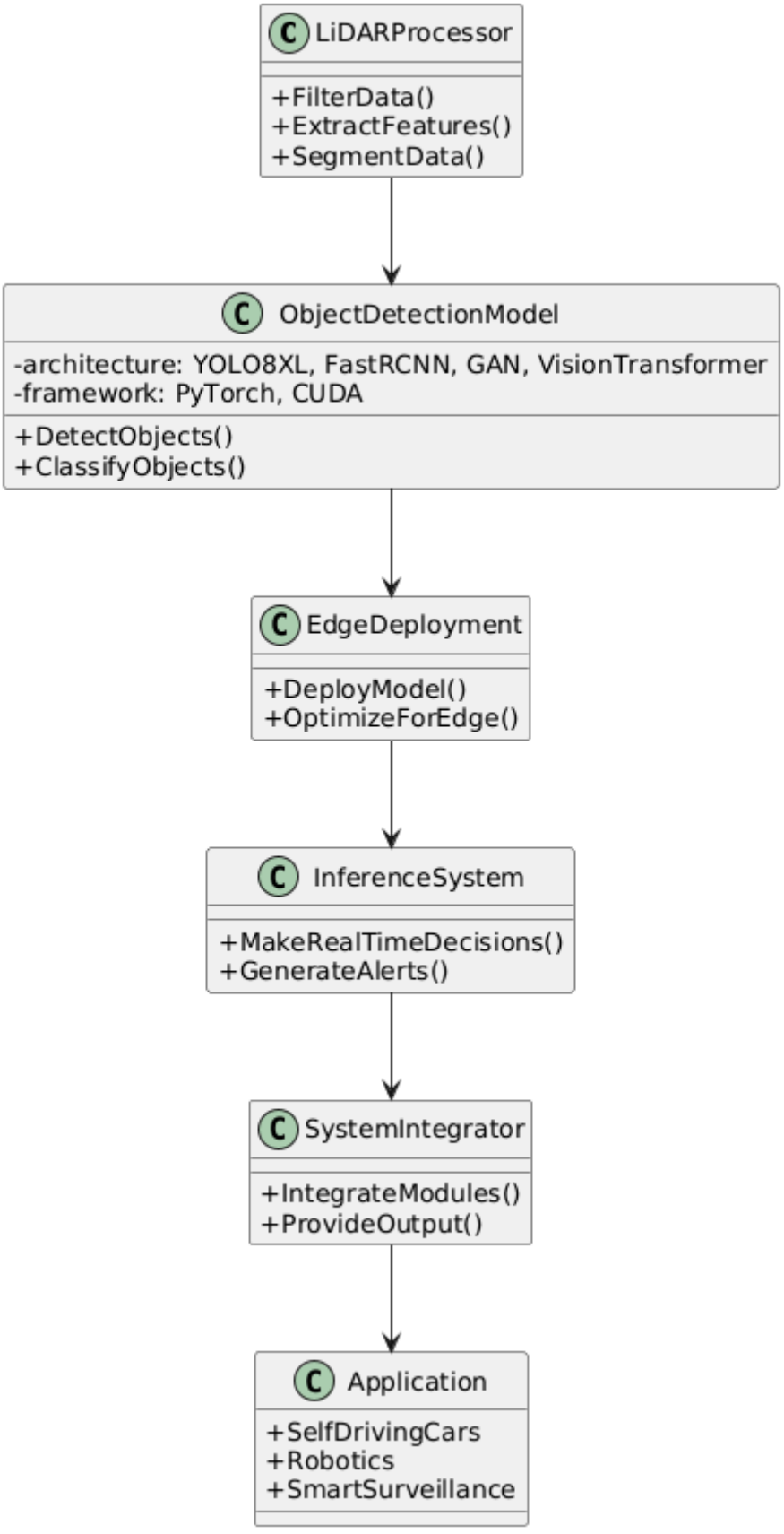

Figure 5-2: Class Diagram

5.2.2 Use case Diagram

A use case diagram is a visual depiction in UML of the interactions between external systems or actors and a specific system. Its primary objective is to display the various actions or functionalities that the system can perform, and how external entities can interact with these actions. The use case diagram can help during the requirements gathering and analysis phase to identify the functional requirements of the system.

The following is the use case diagram of the system:

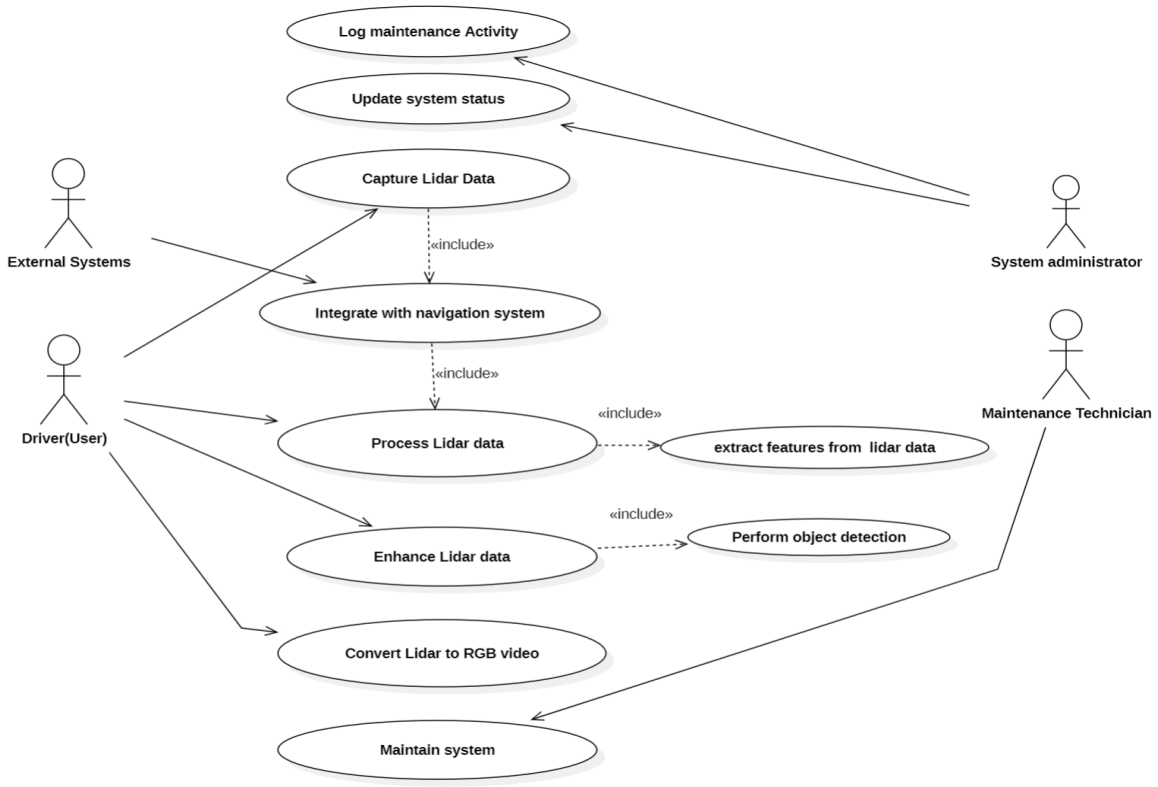

Figure 5-3: Use Case Diagram

5.2.1 Sequence Diagram

A sequence diagram, otherwise called a system sequence diagram (SSD), shows process collaborations in time order in the domain of computer programming. It portrays the locked-in cycles and the messages passed between them to achieve functionality.

The following represents the sequence diagram:

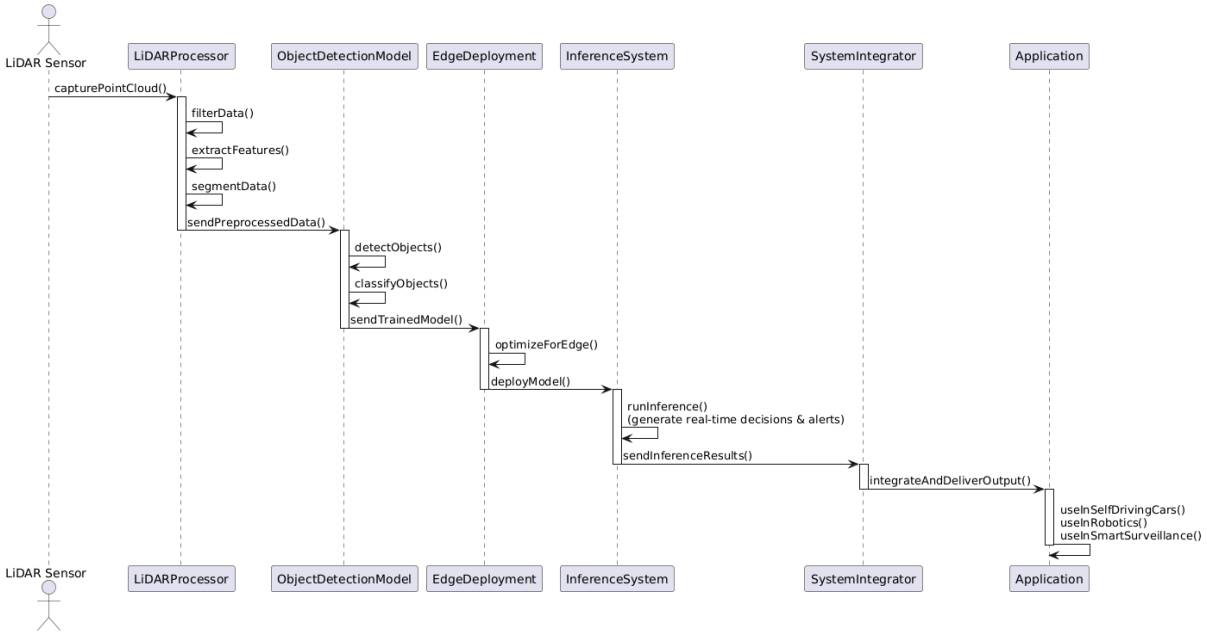

Figure 5-4: Sequence Diagram

CHAPTER 6: METHODOLOGY

6.1 Data Collection and Annotation

There are existing benchmark datasets like KITTI, Waymo, NYU Landscapes etc, that can be utilized for the training purposes. These datasets contain raw LiDAR data points which are annotated with 3D object labels

6.2 Point Cloud Representation

To feed the point cloud data to the model a LiDAR point cloud representation is needed. Different methods like Voxelization (converting points into a 3D Voxel or a grid), BEV (converting the 3D point clouds into a 2D representation to reduce complexity). Some methods use raw points to preserve spatial information.

6.3 Training

A pre-trained model from OpenPCDet frameworks like the PointPillars is trained accordingly on the custom dataset created. The model is optimized for both speed and accuracy.

6.4 Testing and Performance Evaluation

Upon completion of training, the performance of the trained model is rigorously tested across diverse scenarios. Various performance metrics like the mAP (Mean Average Precision), IoU (Intersection over Union) are used to test the model.

6.5 Integration with Object Detection Pipeline

The trained and fine-tuned model is seamlessly integrated into the Object detection pipeline. The LiDAR camera Intel Realsense L515 sends the LiDAR data throught it's rs-capture API which is then processed and sent to the detection model.

6.6 Post-processing Techniques

To enhance the model's accuracy, post-processing steps are applied to output data after feeding it into the model. Techniques like NMS (Non Max Suppression) and Bounding Box refinement are used to remove overlapping detections and to improve detection score respectively.

6.7 User Interaction and Personalized Recommendations

User interaction is seamlessly integrated into the system through dedicated applications such as the infotainment system in vehicles. Applications can be used to view the output of the detection model in cases of reduced visibility due to weather conditions, where the driver can be spatially aware with the help of this system.

CHAPTER 7: IMPLEMENTATION

7.1 Dataset Preparation

The dataset used for training and evaluating the Smart Parking Lot system was derived from multiple open datasets like KITTI, Waymo, nuScenes and NYU Landscape Dataset. These datasets contain LiDAR Data in different representations like Voxels, Grids, BEV (Bird-Eye-View), Pillars etc. Often the Open Datasets provide data in the form of raw points which are used to spatially encode maximum information in the 3D space. However this data needs to be processed for decreasing the complexity and to increase the accuracy during object detection.

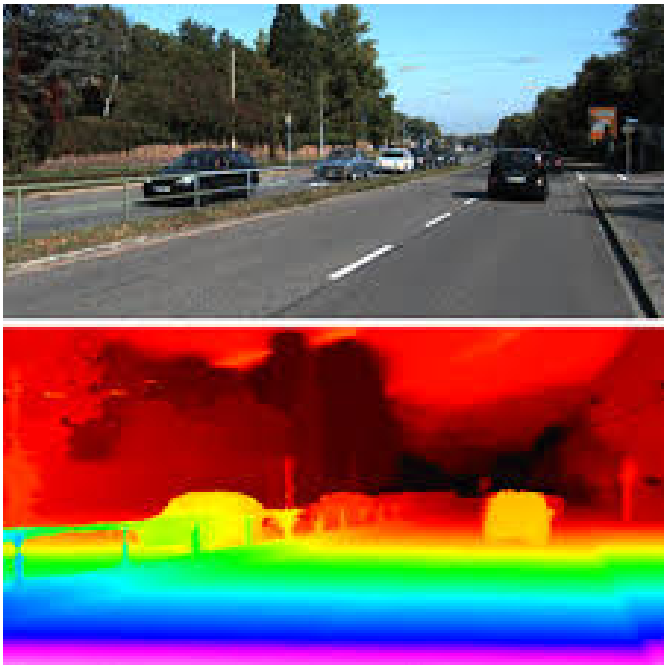

figure 7-1: sample input data.

7.2 Preprocessing

To address the challenges posed by the processing and object detection on raw point clouds of LiDAR Data, pre-processing techniques like LiDAR Data Representaion, 3D Space De-Noising, etc are performed.

Raw point clouds are transformed to various structured data to preserve maximum spatial information while optimizing the processing speed and complexity of the model. Common Represntations include :

1. **Voxels :** The 3D space is divided into smaller cubes/voxels that aggregates all the points in within it. This is done to process the data efficiently.
2. **BEV (Bird Eye View) :** The 3D space is mapped onto a 2D top-down plane to reduce computational complexity.
3. **Pillars :** The pillars are an efficient way to aggregate the points of a space into pillars/vertical columns
4. **Graphs :** In some advanced models, data is mapped to graphs where each point represents a node that is connected to the rest based on spatial proximity.

Noise in the Data can be filtered out using statistical techniques like Ground Plane Removal, Range Filtering, outlier removal.

7.3 Training Deep Learning Model

In this project we used a deep learning model pre-trained on KITTI custom data to demonstrate LiDAR Object Detection. The training pipeline involves the following steps

7.3.1 Model Architecture - PointPillars

We used the **PointPillars** architecture, a single-stage 3D object detection model designed for real-time inference. It converts point clouds into 2D pseudo-images using vertical pillar encoding and applies 2D convolutional layers for detection. It is faster and thus is desirable in cases where inference speeds are the priority.

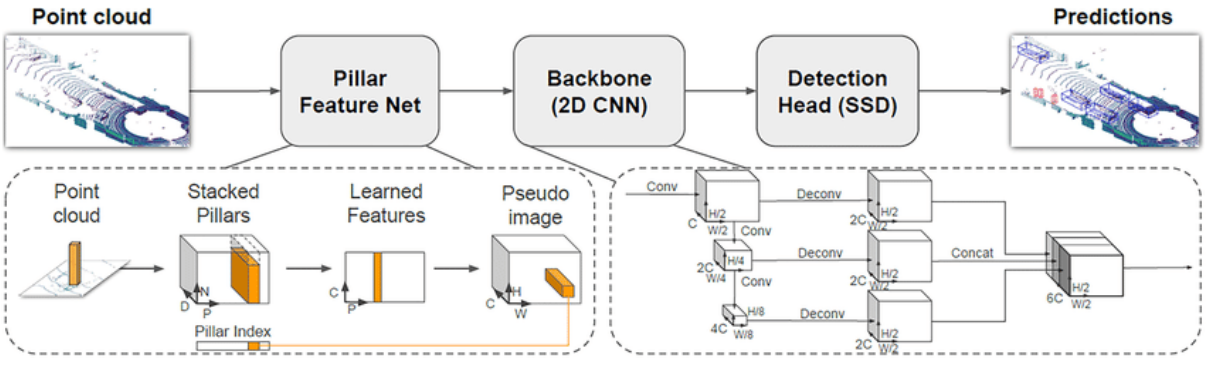

figure 7-2: model architecture.

7.3.2 Loss Functions

The Model trained is used for Detection purposes and hence the following loss functions are used,

1. Cross Entropy (Classification Loss)
2. Smooth L1 (Localization Loss) [Fort Bounding Boxes and localizing the objects]
3. Orientational/Directional Loss

7.3.3 Pre-Trained Model Use

For this project, we used a pre-trained model of PointPillars on KITTI Dataset from the OpenPCDet Framework this enabled us to skip resource-intensive tasks like, training from scratch with zero weights and validate the model perfromance

7.4 Postprocessing

After the Deep Learning model predicts the objects in an image with bounding boxes, post-processing steps are applied to refine the output. Steps like Non-Max-Suppression (NMS), Confidence Thresholding, Bounding Box Decoding, Label Mapping etc are performed to refine the detection results.

7.5 Integration into Pipeline

After the model is completely trained and tested it is integrated into the detection pipeline that consists of a Micro-Controller Board like Jetson Nano, where the model runs. The LiDAR Data is provided by the Intel Realsense LiDAR L515 which has a range upto 10 m. A screen would be showing all the detection results with the bounding boxes around the objects.

CHAPTER 8: RESULTS

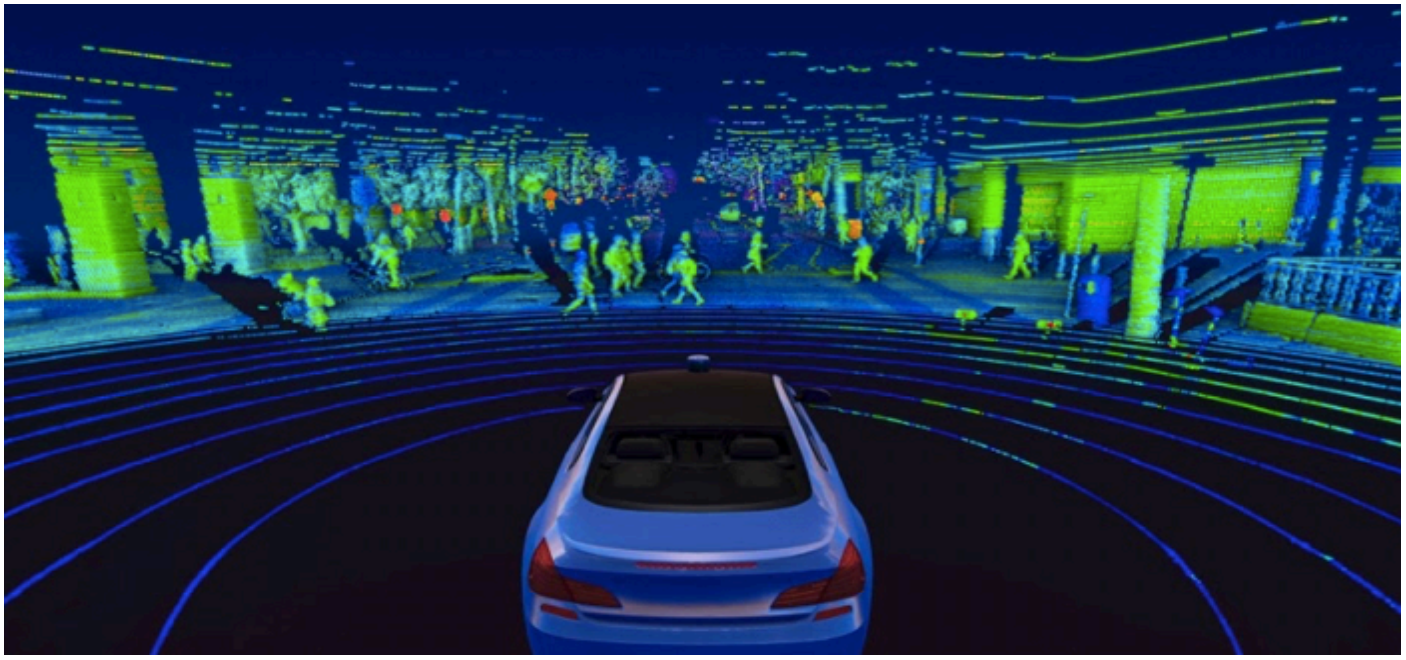

Figure 8-1: Input image to the model.

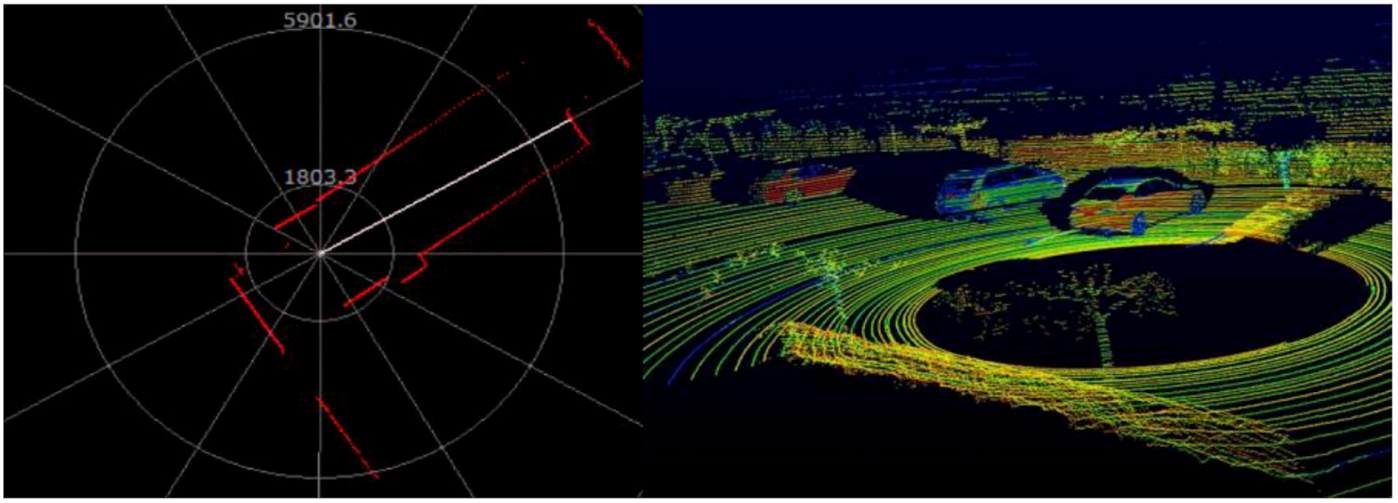

Figure 8-2 : spatial mapping of environment.

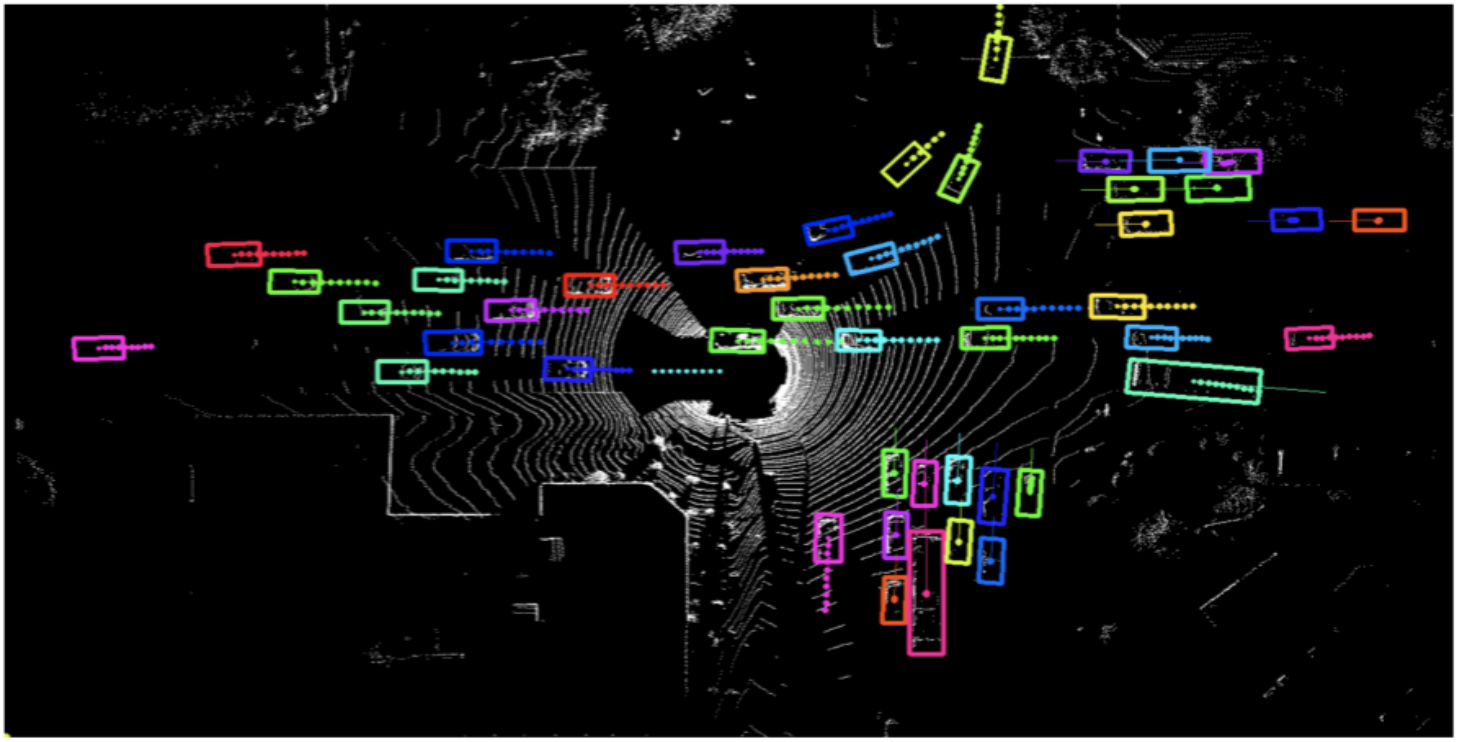

Figure 8-3: BEV(Bird's Eye View) object detection.

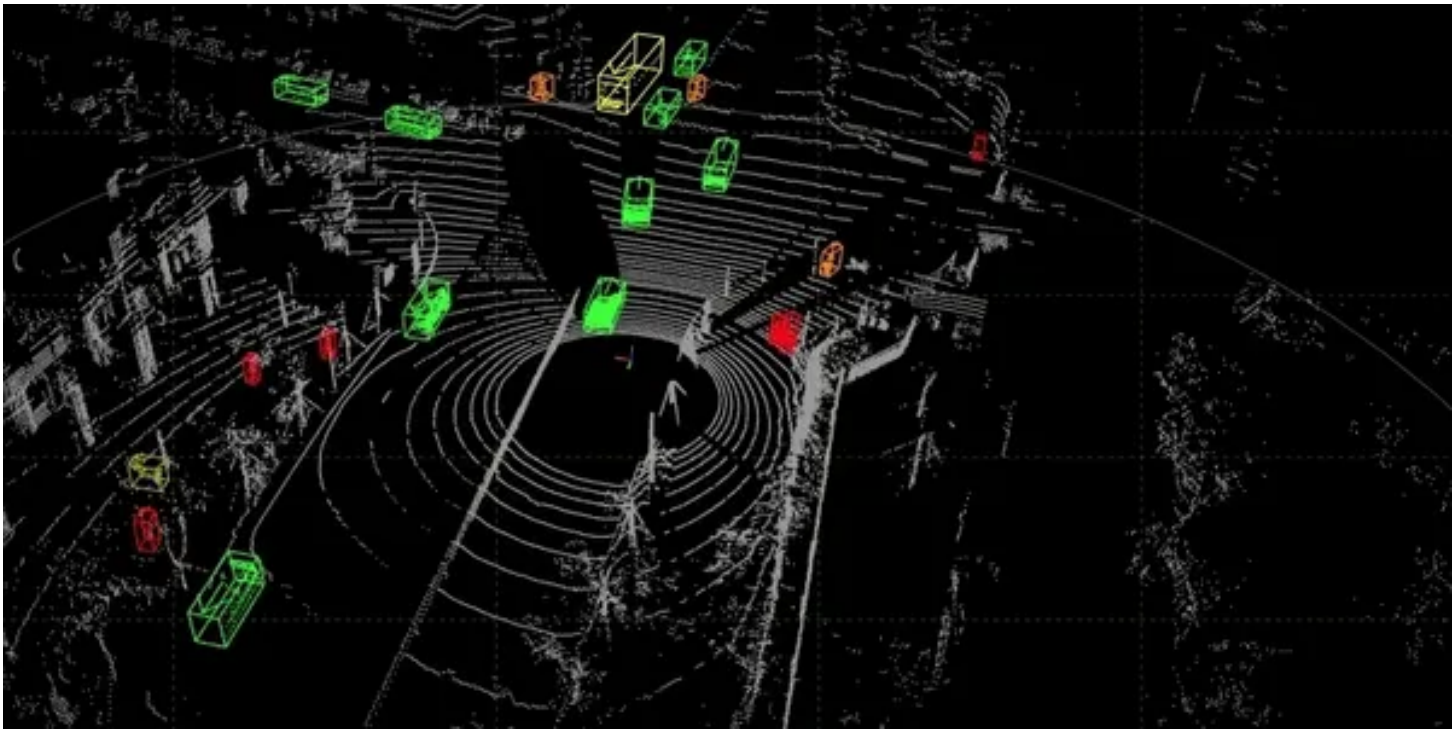

Figure 8-4: Final detection of objects.

CHAPTER 9: CONCLUSION

The 3D detection with LiDAR project demonstrates that the use of deep learning architectures in combination with the traditional LiDAR data processing techniques can significantly improve the speed and accuracy of the model's predictions. By using a point pillar architecture and leveraging open source tools like OpenPCDet, the system can operate with minimal setup allowing it to be deployed in end-user devices and applicable across various situations. Visualizing the results using the Bird's Eye View provides better understanding of the models' performance and its spatial awareness. The overall implementation of this model highlights the potential of LiDAR based perception systems in diverse applications such as autonomous driving, robotics, drones, traffic monitoring, and many more.

CHAPTER 10: FUTURE SCOPE

This section explores potential avenues for further development of the proposed system.

- The accuracy of uncertainty estimation of the model could be enhanced by conducting further research, exploring different probabilistic models or incorporating additional contextual information from the environment.
- The model could be assessed using real world data wherein the model will be put through a variety of weather conditions, varying lighting environments, etc. This would help in understanding the generalizability of the model.
- The possibility of adding a temporal dimension to the model could be researched which could improve the accuracy of detection in dynamic environments such as a city road with moving vehicles, etc.
- Transfer learning ability of the model could be explored so as to extend the use of these models to various other domains which have limited training data.

CHAPTER 11: BIBLIOGRAPHY

[1] Z. Liu, H. Tang, A. Amini, X. Yang, H. Mao, D. L. Rus, S. Han, Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation (2023) 2774–2781 doi:10.1109/ICRA48891. 2023.10160968.

[2] J. Deng, S. Shi, P.-C. Li, W. gang Zhou, Y. Zhang, H. Li, Voxel r-cnn: Towards high performance voxel-based 3d object detection, ArXiv abs/2012.15712 (2020).

URL https://api.semanticscholar.org/ CorpusID:229923684

[3] Y. Wu, Y. Wang, S. Zhang, H. Ogai, Deep 3d object detection networks using lidar data: A review, IEEE Sensors Journal 21 (2) (2021) 1152– 1171. doi:10.1109/JSEN.2020.3020626.

[4] A. Geiger, P. Lenz, R. Urtasun, Are we ready for autonomous driving? the kitti vision benchmark suite, in: 2012 IEEE Conference on Computer Vision and Pattern Recognition, 2012, pp. 3354–3361. doi:10.1109/ CVPR.2012.6248074.

[5] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, O. Beijbom, nuscenes: A multimodal dataset for autonomous driving (2020). arXiv:1903.11027. URL https://arxiv.org/abs/1903.11027

[6] S. Ettinger, S. Cheng, B. Caine, C. Liu, H. Zhao, S. Pradhan, Y. Chai, B. Sapp, C. Qi, Y. Zhou, Z. Yang, A. Chouard, P. Sun, J. Ngiam, V. Vasudevan, A. McCauley, J. Shlens, D. Anguelov, Large scale interactive motion forecasting for autonomous driving : The waymo open motion dataset (2021). arXiv:2104.10133.

URL https://arxiv.org/abs/2104.10133

[7] I. Maksymova, C. Steger, N. Druml, Review of lidar sensor data acquisition and compression for automotive applications, Proceedings 2 (13) (2018). doi:10.3390/proceedings2130852. URL https://www.mdpi.com/2504-3900/2/13/852

[8] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: Unified, real-time object detection (2016) 779–788doi:10.1109/CVPR.2016.91.

[9] X.-F. Han, J. S. Jin, M.-J. Wang, W. Jiang, L. Gao, L. Xiao, A review of algorithms for filtering the 3d point cloud, Signal Processing: Image Communication 57 (2017) 103–112.

doi:https://doi.org/10.1016/j.image.2017.05.009.

URL https://www.sciencedirect.com/science/ article/pii/S0923596517300930

[10] S. Shi, X. Wang, H. Li, Pointrcnn: 3d object proposal generation and detection from point cloud, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 770–779. doi:10. 1109/CVPR.2019.00086.

[11] Y. Zhou, O. Tuzel, VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection (2018) 4490–4499doi:10.1109/CVPR. 2018.00472.

URL https://doi.ieeecomputersociety.org/10.1109/ CVPR.2018.00472

[12] Y. Zhang, Q. Zhang, Z. Zhu, J. Hou, Y. Yuan, Glenet: Boosting 3d object detectors with generative label uncertainty estimation (2024). arXiv: 2207.02466.

URL https://arxiv.org/abs/2207.02466

[13] Y. Xu, X. Tong, U. Stilla, Voxel-based representation of 3d point clouds: Methods, applications, and its potential use in the construction industry, Automation in Construction 126 (2021) 103675. doi:https://doi.org/10.1016/j.autcon.2021.103675.

URL https://www.sciencedirect.com/science/ article/pii/S0926580521001266

[14] Y. Hu, H. You, Z. Wang, Z. Wang, E. Zhou, Y. Gao, Graph-mlp: Node

classification without message passing in graph (2021). arXiv:2106. 04051.

URL https://arxiv.org/abs/2106.04051

[15] Q. Xu, Y. Zhong, U. Neumann, Behind the curtain: Learning occluded shapes for 3d object detection (2021). arXiv:2112.02205.

URL https://arxiv.org/abs/2112.02205

[16] Z. Xia, X. Pan, S. Song, L. E. Li, G. Huang, Vision transformer with deformable attention (2022). arXiv:2201.00520.

URL https://arxiv.org/abs/2201.00520

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need (2023). arXiv: 1706.03762.

URL https://arxiv.org/abs/1706.03762

[18] X. Bai, Z. Hu, X. Zhu, Q. Huang, Y. Chen, H. Fu, C.-L. Tai, Transfusion: Robust lidar-camera fusion for 3d object detection with transformers (2022). arXiv:2203.11496.

URL https://arxiv.org/abs/2203.11496

[19] T. Yin, X. Zhou, P. Kr¨ahenb¨uhl, Center-based 3d object detection and tracking (2021). arXiv:2006.11275.

URL https://arxiv.org/abs/2006.11275

[20] J. Richter, F. Faion, D. Feng, P. B. Becker, P. Sielecki, C. Glaeser, Understanding the domain gap in lidar object detection networks (2022). arXiv:2204.10024.

URL https://arxiv.org/abs/2204.10024

[21] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, O. Beijbom, Pointpillars: Fast encoders for object detection from point clouds (2019). arXiv:1812.05784.

URL https://arxiv.org/abs/1812.05784

[22] S. Shi, C. Guo, L. Jiang, Z.Wang, J. Shi, X.Wang, H. Li, Pv-rcnn: Pointvoxel feature set abstraction for 3d object detection (2021). arXiv: 1912.13192.

URL https://arxiv.org/abs/1912.13192

[23] S. Shi, L. Jiang, J. Deng, Z. Wang, C. Guo, J. Shi, X. Wang, H. Li, Pvrcnn++: Point-voxel feature set abstraction with local vector representation for 3d object detection (2022). arXiv:2102.00463.

URL https://arxiv.org/abs/2102.00463

[24] Z. Liu, H. Tang, Y. Lin, S. Han, Point-voxel cnn for efficient 3d deep learning (2019). arXiv:1907.03739.

URL https://arxiv.org/abs/1907.03739

[25] H. Hu, F. Wang, J. Su, Y. Wang, L. Hu, W. Fang, J. Xu, Z. Zhang, Ea-lss: Edge-aware lift-splat-shot framework for 3d bev object detection, 2023.

URL https://api.semanticscholar.org/CorpusID:257900888

[26] R. Q. Charles, H. Su, M. Kaichun, L. J. Guibas, Pointnet: Deep learning on point sets for 3d classification and segmentation, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 77–85. doi:10.1109/CVPR.2017.16.

[27] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, in: Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS'17, Curran Associates Inc., Red Hook, NY, USA, 2017, p. 6000–6010.

[28] Y. Jiao, Z. Jie, S. Chen, J. Chen, L. Ma, Y.-G. Jiang, Msmdfusion: Fusing lidar and camera at multiple scales with multi-depth seeds for 3d object detection, in: 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2023, pp. 21643–21652. doi: 10.1109/CVPR52729.2023.02073.

[29] Y. Qin, C.Wang, Z. Kang, N. Ma, Z. Li, R. Zhang, Supfusion: Supervised lidar-camera fusion for 3d object detection, in: 2023 IEEE/CVF International Conference on Computer Vision (ICCV), 2023, pp. 21957–21967. doi:10.1109/ICCV51070.2023.02012.

CHAPTER 12: SOURCE CODE

```python
# This script captures depth data from a RealSense camera, processes it using a PointPillars model,
import numpy as np
import pyrealsense2 as rs
import torch
from pcdet.models import build_network, load_data_to_gpu # type: ignore
from pcdet.config import cfg, cfg_from_yaml_file # type: ignore
from pcdet.datasets.kitti.kitti_dataset import KittiDataset # type: ignore
from pcdet.utils import common_utils # type: ignore
import open3d as o3d

# Load model config and checkpoint
cfg_file = 'tools/cfgs/kitti_models/pointpillars.yaml'
ckpt_file = 'F:/models/pointpillars.pth'  # path to the pretrained model
cfg_from_yaml_file(cfg_file, cfg)
logger = common_utils.create_logger()
model = build_network(model_cfg=cfg.MODEL, num_class=len(cfg.CLASS_NAMES), dataset=KittiDataset(cfg=cfg, class_names=cfg.CLASS_NAMES, training=False, root_path=None, logger=logger))
model.load_params_from_file(filename=ckpt_file, logger=logger, to_cpu=False)
model.cuda().eval()

# RealSense pipeline setup
pipeline = rs.pipeline()
config = rs.config()
config.enable_stream(rs.stream.depth, 1024, 768, rs.format.z16, 30)
pipeline.start(config)
```

```python
def convert_depth_to_points(depth_frame, intrinsics):
    depth_image = np.asanyarray(depth_frame.get_data())
    h, w = depth_image.shape
    fx, fy = intrinsics.fx, intrinsics.fy
    cx, cy = intrinsics.ppx, intrinsics.ppy
    points = []

    for y in range(h):
        for x in range(w):
            z = depth_image[y, x] * 0.001  # convert mm to meters
            if z == 0:
                continue
            x3d = (x - cx) * z / fx
            y3d = (y - cy) * z / fy
            points.append([x3d, y3d, z])
    return np.array(points, dtype=np.float32)

try:
    while True:
        frames = pipeline.wait_for_frames()
        depth = frames.get_depth_frame()
        if not depth:
            continue
        intrinsics = depth.profile.as_video_stream_profile().intrinsics
        point_cloud = convert_depth_to_points(depth, intrinsics)

        # Create fake KITTI-style input dict
        input_dict = {
            'points': torch.tensor(np.hstack((np.zeros((point_cloud.shape[0], 1)), point_cloud)), dtype=torch.float32).cuda().unsqueeze(0)
        }
        load_data_to_gpu(input_dict)
        with torch.no_grad():
            pred_dicts, _ = model(input_dict)

        print(pred_dicts[0])  # Print predictions (boxes, scores, labels)

        # Visualization (optional)
        pc = o3d.geometry.PointCloud()
        pc.points = o3d.utility.Vector3dVector(point_cloud)
        o3d.visualization.draw_geometries([pc])

except KeyboardInterrupt:
    pipeline.stop()
```

```python
import streamlit as st # type: ignore
from inference import run_pointpillars_inference # type: ignore
from utils_realsense import get_depth_frame # type: ignore

st.set_page_config(page_title="PointPillars RealSense Demo", layout="wide")
st.title("RealSense + PointPillars 3D Object Detection")

conf_threshold = st.sidebar.slider("Confidence Threshold", 0.0, 1.0, 0.5)

st.subheader(" Simulated Depth Frame")
depth_frame = get_depth_frame()
st.image(depth_frame, caption="Simulated RealSense Depth Frame", use_column_width=True)

if st.button("Run PointPillars Inference"):
    with st.spinner("Running inference..."):
        detections = run_pointpillars_inference(conf_threshold)
        st.success("Inference complete!")

    st.subheader(" Detections")
    for det in detections:
        st.write(f"Object: {det['class']} | Confidence: {det['score']}")
```

```python
import open3d as o3d
import numpy as np

def generate_fake_point_cloud():
    """Simulate a 3D point cloud (as RealSense would provide)."""
    pcd = o3d.geometry.PointCloud()
    points = np.random.rand(1000, 3)  # 1000 random 3D points
    pcd.points = o3d.utility.Vector3dVector(points)
    return pcd

def visualize_point_cloud(pcd):
    """Visualize point cloud using Open3D."""
    o3d.visualization.draw_geometries([pcd])


from visualize_open3d import generate_fake_point_cloud, visualize_point_cloud # type: ignore

pcd = generate_fake_point_cloud()
visualize_point_cloud(pcd)
```

Report

ORIGINALITY REPORT

19%	14%	11%	5%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	arxiv.org Internet Source	3%
2	link.springer.com Internet Source	2%
3	arkajainuniversity.ac.in Internet Source	1%
4	www.coursehero.com Internet Source	1%
5	Submitted to Arab Open University Student Paper	1%
6	Ziying Song, Haiyue Wei, Caiyan Jia, Yongchao Xia, Xiaokun Li, Chao Zhang. "VP-Net: Voxels as Points for 3D Object Detection", IEEE Transactions on Geoscience and Remote Sensing, 2023 Publication	1%
7	krex.k-state.edu Internet Source	1%
8	Submitted to South Bank University Student Paper	<1%
9	Thangaprakash Sengodan, Sanjay Misra, M Murugappan. "Advances in Electrical and Computer Technologies", CRC Press, 2025 Publication	<1%
10	web.archive.org Internet Source	<1%

11	Ardith D. Bravenec, Karen D. Ward. "Interactive Python Notebooks for Physical Chemistry", Journal of Chemical Education, 2022 Publication	<1 %
12	researchspace.ukzn.ac.za Internet Source	<1 %
13	www.slideshare.net Internet Source	<1 %
14	"Pattern Recognition and Computer Vision", Springer Science and Business Media LLC, 2024 Publication	<1 %
15	Huijuan Wang, Xinyue Chen, Quanbo Yuan, Peng Liu. "A review of 3D object detection based on autonomous driving", The Visual Computer, 2024 Publication	<1 %
16	Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, Hongsheng Li. "PV-RCNN: Point-Voxel Feature Set Abstraction for 3D Object Detection", 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020 Publication	<1 %
17	api.deepai.org Internet Source	<1 %
18	"Computer Vision – ECCV 2024", Springer Science and Business Media LLC, 2025 Publication	<1 %
19	Yang Jiao, Zequn Jie, Shaoxiang Chen, Jingjing Chen, Lin Ma, Yu-Gang Jiang. "MSMDFusion: Fusing LiDAR and Camera at Multiple Scales	<1 %

with Multi-Depth Seeds for 3D Object Detection", 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2023
Publication

20	Jia Qin, Ruizhi Sun, Kun Zhou, Yuanyuan Xu, Banghao Lin, Lili Yang, Zhibo Chen, Long Wen, Caicong Wu. "Lidar-Based 3D Obstacle Detection Using Focal Voxel R-CNN for Farmland Environment", Agronomy, 2023 Publication	<1 %
21	export.arxiv.org Internet Source	<1 %
22	www.semanticscholar.org Internet Source	<1 %
23	Fanyue Sun, Guoxiang Tong, Yan Song. "Efficient flexible voxel-based two-stage network for 3D object detection in autonomous driving", Applied Soft Computing, 2024 Publication	<1 %
24	grietinfo.in Internet Source	<1 %
25	Submitted to University System of Georgia (USG) Student Paper	<1 %
26	Submitted to King Abdullah University of Science and Technology (KAUST) Student Paper	<1 %
27	Submitted to University of Technology, Sydney Student Paper	<1 %
28	impa.usc.edu Internet Source	<1 %

29	www.mdpi.com Internet Source	<1%
30	Submitted to K. J. Somaiya College of Engineering Vidyavihar, Mumbai Student Paper	<1%
31	Markus Roth, Dominik Jargot, Dariu M. Gavrila. "Deep End-to-end 3D Person Detection from Camera and Lidar", 2019 IEEE Intelligent Transportation Systems Conference (ITSC), 2019 Publication	<1%
32	Submitted to UCL Student Paper	<1%
33	Submitted to University of Carthage Student Paper	<1%
34	Xuan He, Fan Yang, Kailun Yang, Jiacheng Lin, Haolong Fu, Meng Wang, Jin Yuan, Zhiyong Li. "SSD-MonoDETR: Supervised Scale-aware Deformable Transformer for Monocular 3D Object Detection", IEEE Transactions on Intelligent Vehicles, 2024 Publication	<1%
35	fastercapital.com Internet Source	<1%
36	www.arxiv-vanity.com Internet Source	<1%
37	"Computer Vision – ECCV 2018", Springer Nature America, Inc, 2018 Publication	<1%
38	Manuel Martínez, Raimar Scherer. "eWork and eBusiness in Architecture, Engineering and Construction", CRC Press, 2020 Publication	<1%

39	par.nsf.gov Internet Source	<1%
40	senior.ceng.metu.edu.tr Internet Source	<1%
41	silo.tips Internet Source	<1%
42	Ken T. Mori, Steven Peters. "SHARD: Safety and Human Performance Analysis for Requirements in Detection", IEEE Transactions on Intelligent Vehicles, 2024 Publication	<1%
43	Yucedag, Onur Can. "Olister: Observing LiDAR-Induced Sources for Transferability, Estimation and Robustness in 3D Object Detection", Michigan State University Publication	<1%
44	api-depositonce.tu-berlin.de Internet Source	<1%
45	ebin.pub Internet Source	<1%
46	etheses.dur.ac.uk Internet Source	<1%
47	ijimai.org Internet Source	<1%
48	vdocuments.mx Internet Source	<1%
49	www.ce.cit.tum.de Internet Source	<1%
50	Mohammad Muntasir Rahman, Yanhao Tan, Jian Xue, Ke Lu. "Recent Advances in 3D Object Detection in the Era of Deep Neural	<1%

	Networks: A Survey", IEEE Transactions on Image Processing, 2020 Publication	
51	Sen, Sabyasachi. "In-Depth Stability Characterization and Engineering of Bacterial N-Terminal Motifs and Their Protective Tags", University of Delaware Publication	<1 %
52	Shaoshuai Shi, Zhe Wang, Jianping Shi, Xiaogang Wang, Hongsheng Li. "From Points to Parts: 3D Object Detection from Point Cloud with Part-aware and Part-aggregation Network", IEEE Transactions on Pattern Analysis and Machine Intelligence, 2020 Publication	<1 %
53	Yingjuan Tang, Hongwen He, Yong Wang, Zan Mao, Haoyu Wang. "Multi-modality 3D object detection in autonomous driving: A review", Neurocomputing, 2023 Publication	<1 %
54	d-nb.info Internet Source	<1 %
55	library2.smu.ca Internet Source	<1 %
56	openaccess.thecvf.com Internet Source	<1 %
57	repository.tudelft.nl Internet Source	<1 %
58	research.tue.nl Internet Source	<1 %
59	ruor.uottawa.ca Internet Source	<1 %

60	Alix Marie d'Avigneau, Lilia Potseluyko, Nzebo Richard Anvo, Hussameldin M. Taha et al. "CAMHighways: The Cambridge Highways dataset", Advanced Engineering Informatics, 2025 Publication	<1 %
61	Ton Duc Thang University Publication	<1 %
62	Yifan Zhang, Qijian Zhang, Zhiyu Zhu, Junhui Hou, Yixuan Yuan. "GLENet: Boosting 3D Object Detectors with Generative Label Uncertainty Estimation", International Journal of Computer Vision, 2023 Publication	<1 %
63	Yingjie Wang, Qiuyu Mao, Hanqi Zhu, Jiajun Deng, Yu Zhang, Jianmin Ji, Houqiang Li, Yanyong Zhang. "Multi-Modal 3D Object Detection in Autonomous Driving: A Survey", International Journal of Computer Vision, 2023 Publication	<1 %
64	Jiageng Mao, Shaoshuai Shi, Xiaogang Wang, Hongsheng Li. "3D Object Detection for Autonomous Driving: A Comprehensive Survey", International Journal of Computer Vision, 2023 Publication	<1 %
65	Shaoshuai Shi, Xiaogang Wang, Hongsheng Li. "PointRCNN: 3D Object Proposal Generation and Detection From Point Cloud", 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019 Publication	<1 %
66	eprints.nottingham.ac.uk Internet Source	

<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		