National College of Ireland

Lidar based 3D Detection of Vehicles and Obstacles in Autonomous Vehicles Using Deep Learning Techniques: Analysis and Comparison

MSc Research Project
MSc in Artificial Intelligence

Hugh Plunkett
Student ID: 23312173

School of Computing
National College of Ireland

Supervisor: Abdul Shahid

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Hugh Plunkett
Student ID:	23312173
Programme:	MSc in Artificial Intelligence
Year:	2025
Module:	MSc Research Project
Supervisor:	Abdul Shahid
Submission Due Date:	11/08/2025
Project Title:	Lidar based 3D Detection of Vehicles and Obstacles in Autonomous Vehicles Using Deep Learning Techniques: Analysis and Comparison
Word Count:	XXX
Page Count:	19

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	September 15, 2025

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

LiDAR based 3D Detection of Vehicles and Obstacles in Autonomous Vehicles Using Deep Learning Techniques: Analysis and Comparison

Hugh Plunkett
Student ID: 23312173

Abstract

Autonomous Vehicles (AVs) face significant challenges in terms of accurate and real-time 3D object detection, which is critical for autonomous driving. This study aims to explore and compare several prominent state-of-the-art deep learning (DL) approaches on the KITTI dataset. Using a multifaceted evaluation approach, we conducted a standardized analysis of the most recent LiDAR-based 3D detectors on the dataset. Our work uniquely integrates hybrid models into the comparison and addresses real-world deployment challenges by evaluating computational trade-offs between speed and accuracy. A thorough comparison of LiDAR-based 3D detectors is needed to assess their accuracy and speed for real-world AV deployment. Among the eight models evaluated, PointRCNN and VoxelRCNN achieved the highest accuracy, followed closely by Complex Yolo v4 and PV-RCNN, indicating strong performance in 3D detection tasks. The results highlight the better performance of voxel-based and point-based deep learning models over traditional approaches in both accuracy (71 and 68 mAP compared to AVOD's 47 mAP) and efficiency (VoxelNet runs at 46 FPS, 3x faster than AVOD while being more accurate). All models have been tested in a standardised environment with the hope to provide valuable insights for selecting models, advancing research toward practical, high-performance perception systems in AVs.

Keywords: Deep Learning Survey, LiDAR 3D Detection, Autonomous Vehicles, Deep Learning for Perception, Point Cloud Processing, Computer Vision

1 Introduction

1.1 Research Motivation

Autonomous vehicles (AVs), often referred to as self-driving cars, are vehicles capable of navigating and functioning with minimal or no human input. Instead, they depend on a mix of sensors, machine learning algorithms, and AI-driven systems. AVs use a perception system known as "Computer Vision" (CV) to navigate safely. Now one of AI's most transformative applications in modern technology, AVs have drawn considerable interest as a promising way to address urban mobility issues, such as traffic congestion [1], while also improving efficiency and public safety as a whole [2].

Due to its efficiency, 2D camera-based object detection is used throughout the AV industry - most notably in the Tesla vehicle "Autopilot driver capability" systems, which provides advanced driver-assistance capabilities, including adaptive cruise control, lane-keeping, and automated lane changes [3]. However, AV systems such as Tesla's which purely use cameras also have several potential flaws and security issues. A key vulnerability is the susceptibility to "phantom attacks", where projected images of fake objects (like pedestrians or signs) can be used to trick the system [4]. These attacks succeed when fake objects exist across multiple frames, tricking object-tracking algorithms and can cause the AV to act unpredictably by suddenly braking, rapidly accelerating or turning into obstacles endangering both occupants of the vehicle and people around them. These vulnerabilities

endanger passengers and pedestrians, highlighting the need for robust alternatives.
These significant limitations demonstrates why 2D camera-only systems struggle with adversarial scenarios, highlighting the need for 3D LiDAR based detection systems to ensure reliability autonomous driving which is obviously a safety focused field. Current systems remain vulnerable to real-world conditions [5] and due to this, research done in this field to find more efficient 3D detection methods is of high importance. This makes our research question a highly relevant and important one.
Recent advancements in deep learning approaches such as PointRCNN [6] and Fast Point RCNN [7] have shown substantial improvements in 3D object detection performance, achieving both faster processing speeds and greater accuracy while maintaining similar resource requirements to previous methods.

1.2 Research Question

The above challenges motivate the following research questions:

> *How do state-of-the-art deep learning techniques compare in accuracy, efficiency, and speed for LiDAR-based 3D object detection in autonomous driving scenarios?*
>
> *How do hybrid LiDAR-based 3D object detection approaches compare to pure projection-based, voxel-based, and raw-point-based methods in terms of accuracy and computational efficiency?*

1.3 Research Objectives

This research aims to explore and compare several prominent state-of-the-art deep learning (DL) techniques for 3D object detection in AVs on KITTI [8] a large publicly available dataset well suited for both birds-eye view (BEV) and 3D object detection problems. These techniques would have also be compatible and commonly used on other large publicly available datasets such as Waymo's open dataset [9], ApolloScape [10], and nuScenes [11], however for the sake of standardise testing and fair evaluation and comparisons this project will use KITTI [8; 12] as the sole dataset and benchmark.
We also aim to provide the results of each model on this identical architecture to allow future researchers to better compare and select the best model for their needs.

1.4 Background

This section aims to provide some background information on LiDAR technology as well as the dataset this project uses.

1.4.1 Sensor Technologies for AV Perception

One of the most critical challenges in AV technology is the accurate and real-time detection of vehicles and stationary obstacles (such as trees, parked cars or buildings). These 3D detection systems use point clouds (from high performance sensors such as LiDAR, radar or multi-camera arrays) to define an object's position, size, and rotation in 3D space [13]. LiDAR plays a major role in AVs by providing high-resolution, real-time spatial data to enhance object detection and navigation accuracy.

While traditional 2D camera-based detection methods offer faster processing speeds, they struggle with occlusion handling and depth perception, as they interpret objects as flat [14]. In contrast, 3D detection systems can more accurately capture depth, enabling precise distinction between objects at different distances but at a significantly higher computational cost. This increased demand for processing power, often requiring dedicated GPUs, and more computationally effiient models remains a key challenge for autonomous vehicle obstacle detection systems [15].

1.4.2 LiDAR

LiDAR (Light Detection and Ranging) sensors work by sending out laser pulses and calculating the time they take to bounce back, enabling precise distance measurements to objects [1]. The collected information forms a point cloud, which consists of 3D coordinates that detail the environment's geometry and reflectivity. Unlike images, point clouds are unstructured and sparse, but they perform more consistently under various conditions, such as poor lighting or adverse weather.
LiDAR can identify objects over 200 meters away, though the point density on objects of the same size diminishes with range. Despite this, LiDAR point clouds are highly effective at capturing accurate object shapes, dimensions, and positions, making them indispensable for 3D object detection and localization in self-driving systems.
3D object detection methods often use sensor fusion techniques to combine data from LiDAR sensors and cameras, combining the strengths of each method. However the camera and LiDAR sensor must be correctly calibrated first to establish a unified spatial frame of reference, allowing them to be used alongside each other. State-of-the-art algorithms such as Complex-Yolo[16] have demonstrated impressive performance on benchmark datasets, such as the KITTI dataset, using LiDAR point clouds.
Despite their ability to provide accurate, in-depth information and record complex characteristics of objects, LiDAR systems have limitations. They are costly compared to other sensors and typically offer a limited field of view, meaning that multiple units are needed for comprehensive coverage such as "birds-eye-view" or BEV systems.

1.4.3 KITTI Dataset

The KITTI dataset [8] is widely used for autonomous driving research in areas such as 3D object detection. It consists of point cloud data for both BEV and 3D object detection, as well as colour images and camera calibration and label data. This is the primary dataset of this project and was chosen due to its common use as a benchmark in evaluating new detection models, especially multi-class due to its detailed annotations for multiple object classes namely vehicles, pedestrians, and cyclists.
It contains 7,481 training images and point clouds and 7,518 test images and point clouds, covering the three object classes mentioned above. For each class, detection outcomes are evaluated based on three difficulty levels: easy, moderate, and hard, which are determined according to several factors such as the objects size, occlusion state, and truncation level.

[1] Tailte Éireann, "What is LiDAR?," *Tailte Éireann Blog*. [Online]. Available: `https://www.tailte.ie/en/blog/what-is-lidar-.html`. Accessed: 29/03/2025.

1.4.4 Terminology

The following terminology subsection defines key terms, labels, and metrics used in the KITTI 3D object detection dataset. This aims to allow readers and new researchers to familiarise themselves with important details and to ensure consistent interpretation of our results.

- 3D Detection Difficulty: As shown in Figure 1 the KITTI benchmark has three difficulty standards depend on bounding box height, occlusion and truncation levels.
- Intersection over Union (IoU): is TP/(TP+FP+FN), where TP, FP, and FN are the numbers of true positive, false positive, and false negative pixels, respectively.
- 3D AP: Average Precision for 3D object detection, computed using 3D IoU thresholds across three difficulty levels.
- 3D mAP: Mean Average Precision across all classes, providing a single performance metric for 3D detection models on the KITTI benchmark.

$$\text{mAP} = \frac{1}{9}\begin{pmatrix} \text{AP}_{\text{car, easy}} + \text{AP}_{\text{car, moderate}} + \text{AP}_{\text{car, hard}} + \\ \text{AP}_{\text{pedestrian, easy}} + \text{AP}_{\text{pedestrian, moderate}} + \text{AP}_{\text{pedestrian, hard}} + \\ \text{AP}_{\text{cyclist, easy}} + \text{AP}_{\text{cyclist, moderate}} + \text{AP}_{\text{cyclist, hard}} \end{pmatrix} \quad (1)$$

  We also include 3D mAP (Car) and 3D mAP (Easy) to evaluate models on those Car or Easy entries.

- Frames per Second (FPS): Measures the inference speed of a 3D object detection model on the KITTI benchmark, indicating real-time performance capability, with larger numbers indicating better performance.

	Min. bounding box height	Max. occlusion level	Max. truncation
Easy	40px	Fully visible	15%
Moderate	25px	Partly occluded	30%
Hard	25px	Difficult to see	50%

Figure 1: KITTI dataset difficulty level classification standard [17]

.

1.5 Thesis Structure

This paper will surveys existing 3D object detection methods in the Literature Review, categorizing state-of-the-art approaches (projection, voxel, point, and hybrid-based) while identifying gaps in comparative evaluations. The Methodology section outlines

the research design, including model selection, dataset, evaluation metrics, and hardware/software configurations for standardized testing. Results present quantitative performance comparisons, namely accuracy and speed, and introduce the M-Score metric to assess real-world usability. The Discussion section interprets key findings, examines trade-offs between accuracy and efficiency, and evaluates implications for autonomous driving systems. Finally, the Conclusion summarizes contributions, acknowledges limitations and discusses future work.

2 Literature Review

2.1 A Review of Existing Surveys

Due to the rapid growth of deep learning (DL) in computer vision, object detection has become an extensively studied application [14]. Three-dimensional detection draws much attention in robotics applications, such as autonomous driving, because the intention to know object driving environment and location has increased. Most review papers present general detection in 2D and 3D and for multiple sensors, including cameras, radar and LiDAR [15]. Different papers also have different categories that the divide models up into such as voxel-based vs raw point cloud. There are also multiple valid methods of evaluating and comparing these methods, as well as showing the comparisons. This literature review aims to discuss all these considerations as well as identify the research niche.

2.2 A Comparison of 2D Detection Methods and 3D Detection Methods

In the 2019 paper [18] L. Jiao et. al. Goes into great detail providing an overview of 2D object detection methods and splits them into two categories: one-stage (e.g. YOLO) and two-stage detectors (e.g. Faster RCNN). However, while the authors do briefly discuss LiDAR and 3D object detection when discussing trends in the field, the majority of the material that this survey covered falls under 2D detection, which unfortunately does significantly limit its relevance for modern perception tasks. In comparison the 2022 survey [19] carried out by S.Y. Alaba fully evaluates and compares the 3D methods of detection such as voxel-based methods as well as their advantages and disadvantages in terms of accuracy and computational efficiency, which is then backed up by comparisons on multiple autonomous driving datasets (e.g., NuScenes, Waymo). Our paper aims to comprehensively review LiDAR 3D object-detection models for autonomous driving due to 2D models becoming outdated by current standards.
While the 2D methods used discussed in [18] are outdated by today's standards, the contrast between the 2D methods and 3D methods from [19] reflects the field's pivot toward 3D perception and the current demand for perception systems with better depth perception and accuracy as addressed by S. Alaba [19]. Furthermore in the survey [20] the authors critique 2D object detection methods such as those discussed in [18] claiming that 2D object detection does not meet the need of real-world scenarios. This is significant to the field of 3D object detection. However while [19] delivers analysis of LiDAR based 3D detection methods, its fails to give hybrid or sensor-fusion approaches (e.g., LiDAR-camera fusion based methods such as BEVFusion) significant focus. This limits its practical relevance in the field of AV development, in which these hybrid techniques play a major role. This paper aims to extend upon [19] by evaluating the most recent

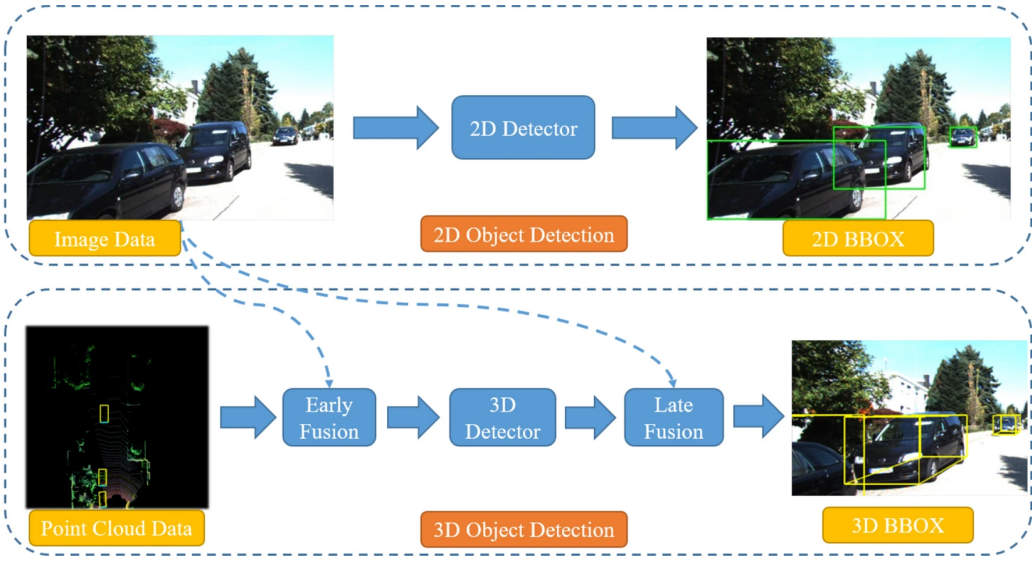

Figure 2: Comparison between 2D and 3D object detection methods [20]

.

methods that have emerged as state-of-the-art since its 2022 publication date. We also want to see how hybrid approaches compare in terms of our research question, as well as examining how different algorithms compare in an identical testing environment.

2.3 Methods of Categorizing Algorithms

S.Y Alaba [19] splits up the models discussed into the following categories: projection based, voxel-based approaches and raw point cloud methods, as does Y. Wu et al in [21] leading to this becoming a common method of categorising in recent literature. As mentioned above a weakness of [21] is the exclusion of hybrid methods, such as those discussed in [22] and [23], something that our research aims to address.

In comparison to [19] and [21], the survey by L. Chen [24] takes a more unique method of categorising, separating models depending on their historical context as part of a evolutionary review. The authors sperate models into 3 categories: Traditional Computer vision algorithms, Deep Learning-based Computer Vision Algorithms and Visual SLAM Algorithms. While this way of separating algorithms provides valuable historical information it lacks modern relevance due to the focus given to historical algorithms that are no longer state of the art. This leads our paper to focus on the more relevant current state-of-the-art models such as Fast Point RCNN [7].

In the 2007 survey [25] by F. Tarsha-Kurdi analysis and comparisons are made papers that focused how on LiDAR data is used for object detection and reconstruction, sharing overlap with our research question. The authors, similarly to [19] break up these papers by methodology and split them into two categories: Model driven approaches and Data driven. Model-driven methods processes the LiDAR point cloud data and matches features to predefined geometric templates that are stored in the models library after first segmentation is done on them splitting them into features terrain, vegetation, and buildings. While these methods work well for simple structures, this method lacks promise in terms of our research field (AVs) as it would requires large data heavy libraries for obstacles such as other cars with irregular designs to be detected and accurately recognised. As such they will be excluded from our paper.

Data-driven methods directly reconstruct objects through a series of geometric operations (e.g., fitting planes and edges) on raw point clouds, similar to the voxel-based approaches from [19]. However, as the paper critiques, these techniques tend to struggle with noise and gaps in LiDAR data, requiring post-processing and while state-of-the-art for its time surveys focusing on newer more efficient data-driven methods have been published.

2.4 Evaluation Metrics

The most commonly used evaluation metrics in 2D detection competitions (such as nuImages by nuScenes [11]), and 3D detection benchmarks [10], is mean Average Precision (mAP) as well as Intersection over Union (IoU) both of which evaluate models based on their detection accuracy. However, many more evaluation metrics are commonly used in categories such as computational efficiency as well as dataset specific evaluation metrics such as nuScenes Detection Score [11].
While the 2007 [25] study evaluates success purely through geometric accuracy (e.g., alignment error), the 2024 survey [24] introduces energy efficiency as an evaluation metric and aims to balance accuracy and performance with the associated computational cost. This metric is critical for our research question, as we aim to evaluate the efficiency of LiDAR-focused algorithms. While we initially considered metrics such as power consumption (watts), computational operations (FLOPs), and memory usage (MBs), our final assessment focuses on practical runtime performance, specifically, inference times and frames per second (FPS), to provide a direct measure of real-world deployment potential.
[20] included evaluation metrics Average Precision (AP) and IoU both of which are crucial for evaluation the detection accuracy of models. Mean average precision, localisation accuracy, and F1 score as also used alongside True Positive Rates (TPR) and False Positive Rates (FPR). In contrast the paper [21] uses an alternative approach with 2D centre distance thresholds rather than IoU to evaluate detections. This method reduces the impact of object size and orientation on scoring, meaning a better balanced evaluation across different object types. IoU as used in [20] might disadvantage smaller objects like pedestrians or cyclists, so the evaluation metric used in this [21] would provide a more consistent assessment regardless of an object's size or orientation. However, to get around this potential issue we choose smaller IoU threshold values (0.5 instead of 0.7) for pedestrians and cyclists so the models are not biased towards classes with larger volumes such as cars.
The two papers contrast further as they have distinct strategies for overall evaluation of the performance evaluation: while [20] uses separate metrics (AP, Average Orientation Precision, Average Orientation Similarity) that gives detailed information as to how well a model works but can complicate direct model comparisons due to there being so many metrics, [21] introduces the unified nuScenes Detection Score (NDS) that combines mAP with five other metrics into a single value, simplifying benchmarking at the potential cost of masking individual model weaknesses. These two evaluation methods are both valid but with different benefits, in particular [20]s approach suits high precision tasks like autonomous driving with a need for high performance across different methods., whereas the method of [21] works well at grading algorithms on one metric providing a more basic insight for quick comparisons. Due to the high accuracy of [20]s approach it is well suited for our research topic and as part of our research question we will evaluate models on multiple metrics.

2.5 Presentation Methods

In the paper [19] the evaluation metrics are presented in the form of a graph, giving it a distinct advantage in terms of readability over some of the other papers mentioned such as [14], [26] or [15]. In this paper the graph clearly shows the AP of several models on the KITTI dataset split up depending on whether the model was tested on Birdseye view (BEV) or 3D LiDAR data from the dataset. Similarly, while not a survey paper as it presents its own research, [7] presents the performance of other models as part of a graph but with considerably more evaluation metrics displayed than [19]. In particular Time (in seconds) and GPU model are used as metrics. The inclusion of both of these metrics allow for insight to be made as to the efficiency of models as more powerful gpus would normally mean lower run times, and lower run times usually means higher efficiency. However to a reader with low knowledge of GPU hardware the inclusion of GPU as a column just adds to visual clutter making it harder to quickly compare the evaluation metrics. This project aims to improve upon this by using a single GPU for testing, resulting in a standardised and fairer evaluation environment.

2.6 Overview of Literature Review

The literature review examines the evolution of object detection in computer vision, focusing on the transition from 2D to 3D methods, particularly for autonomous driving applications, most recently using LiDAR. Existing surveys, such as those by L. Jiao et al. [18] and S. Alaba [19], compare traditional 2D detectors with modern 3D LiDAR-based approaches. Evaluation metrics like mAP, IoU, and NDS are commonly used to assess detection accuracy and computational efficiency, with some studies incorporating runtime performance and energy efficiency. The review also discusses different categorization frameworks, including projection-based, voxel-based, and evolutionary classifications, while highlighting the increasing importance of hybrid sensor-fusion techniques in autonomous vehicle perception systems.

While existing research has made significant progress in 3D object detection using LiDAR data, Table 1 reveals existing gaps in the literature. Most prior works focus on specific aspects such as computational efficiency ([7], [24]) or multi-metric evaluation ([19], [23]), but none provide a comprehensive analysis combining hybrid models, efficiency comparisons, and standardized testing. Notably, only [26] explores hybrid approaches and not on the task of 3D object detection, while over half of papers overlook key real-world usability factors like balanced performance metrics.

Our paper addresses these limitations by offering a fairer evaluation, integrating hybrid models, computational efficiency analysis, and multi-metric assessment (including accuracy and speed) under standardized testing conditions. This approach not only fills a critical research gap but also provides practical insights for deploying 3D object detection in autonomous driving systems, where both performance and efficiency are essential. By unifying these aspects, our work enables more informed model selection and highlights trade-offs that prior studies have not systematically examined. Table 1 shows a summary of the survey of related works in existing review articles.

Table 1: Comparison of Key Features in Related Works

Reference	Hybrid Models	Comp. Efficiency	Multi-Metric Eval.	LIDAR	3D Object Detection	Standardized Testing
[7]	X	Yes	Yes	Yes	Yes	X
[18]	X	X	X	X	Briefly	X
[19]	X	X	Yes	Yes	Yes	Yes
[20]	X	X	Yes	X	Yes	X
[21]	X	X	X	Yes	Yes	X
[23]	X	X	Yes	Yes	Yes	X
[24]	X	Yes	X	X	Yes	X
[25]	X	X	X	Yes	X	X
[26]	Yes	X	X	Yes	X	X
Our Paper	Yes	Yes	Yes	Yes	Yes	Yes

3 Methodology

3.1 Research Design

3.1.1 Overview of Model Types

LiDAR data's sparse, unstructured format makes it difficult to process, requiring advanced techniques for meaningful analysis. Different methods have been developed to process LiDAR point cloud data: projection based approaches, such as Complex YOLO [16] and AVOD [27], which convert the 3D LiDAR data to 2D representations, voxel-based approaches, such as VoxelNet [23] and Voxel-RCNN [28], which build volumetric grids from the data and raw point cloud methods, such as PointRCNN [6] and Fast Point R-CNN [7], which work directly on these point clouds, and hybrid methods such as CaDNN [29] or PV-RCNN [22] which use a combination of the other methods.
Every single model featured on this project was selected for two reasons:

1. They perform at a state of the art level in their respective approach
2. They are open source or had code freely available for academic use.
   Note: in some cases such as Fast Point RCNN we reached out to research teams to obtain their code for use in this project and valuable discussion.

3.1.2 Type 1: Projection Based Approaches

Projection-based LiDAR 3D object detectors convert raw 3D point clouds into structured 2D representations to leverage efficient 2D CNNs for detection. These methods project sparse 3D points onto a 2D plane, encoding attributes like depth, intensity, or height into channels. For example, BEV-based approaches discretize the scene into a grid, preserving spatial relationships, while range-view projections maintain the LiDAR's native spherical perspective. Once projected, 2D CNNs process the data to predict 3D bounding boxes, often combining multi-view projections to mitigate information loss. Popular methods include PointPillars[30] and Complex-YOLO[16], as they balance speed and accuracy by

avoiding direct point cloud processing.
AVOD: This projection-based method works in two stages: first, a Region Proposal Network (RPN) generates 3D object proposals, and then a second-stage network refines these proposals to predict accurate oriented 3D bounding boxes with classifications. AVOD was chosen as it achieves state-of-the-art performance on the KITTI benchmark [12] and is well optimised with open source code available [27].
Complex YOLO V4: This model is a modified version of the YOLOv2 model that is commonly used for 2D object detection using RGB images, extended for 3D object detection in LiDAR point clouds [16]. Evaluated on the KITTI benchmark, it achieves state-of-the-art accuracy for cars, pedestrians, and cyclists while running significantly faster than competing methods.

3.1.3 Type 2: Voxel Based Approaches

Voxel-based 3D object detection methods, such as the popular VoxelNet[23]convert raw LiDAR point clouds into a structured 3D grid knwon as a voxel and apply 3D convolutions for efficient feature extraction. These methods are commonly used as they balance accuracy and computational efficiency by trading off fine-grained point-level details for faster processing. Voxel based approaches, as well as projection based approaches, are the two most popular types of approaching 3D Object detection on the KITTI dataset due to their apparent better trade-off between accuracy and computational efficiency [8].
VoxelNet: This voxel-based approach is currently state-of-the-art for the task of 3D object detection. It directly processes raw LiDAR point clouds using voxel-based feature encoding (VFE), eliminating manual feature engineering. It performs extremely well on the KITTI dataset [23].
Voxel-RCNN: Voxel R-CNN is an efficient 3D object detection method that uses voxel representations instead of raw point clouds. It achieves high accuracy like point-based methods but runs faster by processing organized voxel data, featuring a specialized Voxel RoI Pooling module to refine object predictions. In its 2020 paper it was shown to achieve high performance on KITTI and Waymo datasets matching top detectors while maintaining real-time speed (25 FPS) [28].

3.1.4 Type 3: Raw Point CLoud Methods

Raw point cloud-based 3D object detection methods directly process unstructured LiDAR point clouds without voxelization or projection, leveraging point-wise feature extraction networks to preserve precise 3D geometry.
PointRCNN: A two-stage 3D detector that first generates proposals directly from raw point clouds and then refines them into canonical coordinates, combining local and global features for accurate detection. It outperforms state-of-the-art methods on KITTI using only point cloud data [6]. All the required code is also highly accessible.
Fast Point R-CNN: This raw point cloud method is a fast two-stage 3D detector that combines voxel-based proposals with point-wise attention fusion for accurate refinement. When published it boasted achieving 15FPS state-of-the-art results on the KITTI dataset under 3D/BEV detection [7].

3.1.5 Type 4: Hybrid Approaches

Hybrid 3D object detection methods combine multiple representation paradigms (such as voxels, raw points, or projections all mentioned above) to leverage their complementary strengths and balancing efficiency, accuracy, and geometric precision.
CADNN: This hybrid model is a monocular 3D object detection method that improves depth estimation by predicting a categorical depth distribution for each pixel, rather than directly regressing depth. The model uses a bird's-eye-view (BEV) projection and a single-stage detector for efficient 3D bounding box prediction. CaDDN achieved 1st place among monocular methods on the KITTI 3D benchmark making it state of the art in its domain [29].
PVRCNN: PVRCNN [22] is a hybrid 3D object detection model that combines the efficiency of voxel-based feature extraction with point-based refinement to achieve fine hallucination. It first processes LiDAR data using a fast 3D voxel CNN, then refines predictions with point-level features from PointNet++ for sharper object boundaries and better accuracy. This balance makes it one of the top-performing methods on benchmarks like KITTI.

3.1.6 Research Design Overview

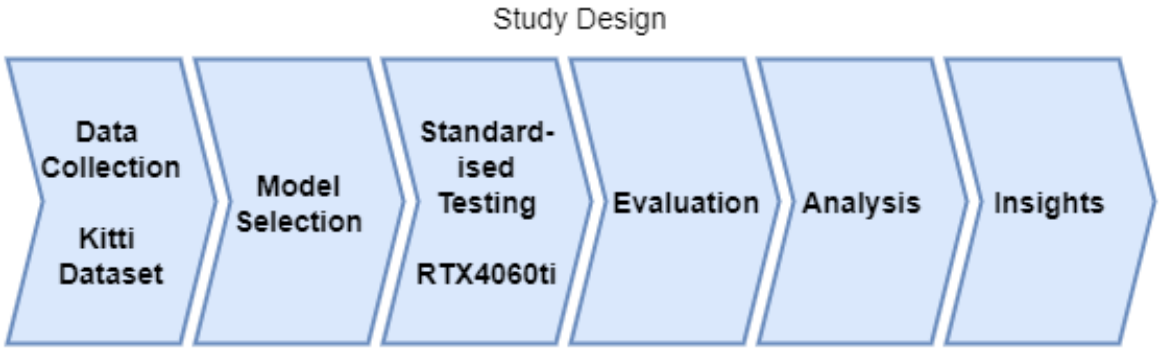

Figure 3: Our research design

This study begins with data collection from the KITTI dataset, comprising LiDAR point clouds and RGB images. Eight state of the art models spanning point based, voxel based, hybrid, and projection based methods are selected for standardized evaluation on identical hardware (RTX 4060 Ti). Performance is measured using accuracy (mAP, class wise AP) and speed (FPS) metrics, with results analysed through a trade off scatter plot. The proposed M Score (balancing mAP and FPS) guides practical insights, such as prioritizing high accuracy models for urban AVs (e.g., PointRCNN) or high speed models for highways (e.g., VoxelNet).

3.2 Data Collection

The KITTI dataset files were downloaded from the official site[2]. In particular the Velodyne point clouds (29 GB), Training labels of object data set (5 MB), Camera calibration matrices of object data set (16 MB) and the Left colour images of object data set (12

[2]`https://www.cvlibs.net/datasets/kitti/eval_object.php?obj_benchmark=3d`

GB).
We then split the data into training and testing using the method described in each models documentation. Usually looking like this:

```
DATA_DIR
├── training   ← training data
│   ├── image_2
│   ├── label_2
│   └── velodyne
└── testing   ← testing data
    ├── image_2
    ├── label_2
    └── velodyne
```

The models themselves were downloaded from their GitHub repositories (such as the open source project OpenPCDet [31]) or shared by researchers upon email requests.

3.3 Analysis Methods

Each of our models were trained on the KITTI training split using their native preprocessing pipelines to ensure optimal performance. For evaluation, we tested all models on the KITTI test split and recorded their 3D Average Precision (AP) scores for the Car, Pedestrian, and Cyclist classes across all three difficulty levels (Easy, Moderate, Hard). Additionally, we measured the inference speed (FPS) of each model on standardized hardware to evaluate computational efficiency.
The models were then systematically compared using four key metrics: overall detection accuracy (mAP across all classes), vehicle-specific performance (mAP for Cars), performance on simple scenarios (mAP for Easy difficulty), and runtime efficiency (FPS). This multi-faceted evaluation approach provides insights into both the detection capabilities (accuracy) and practical deployment potential (efficiency) of contemporary 3D object detection architectures.

3.4 Experimental Setup

3.4.1 Hardware

The benchmarking of all models was done on the following hardware:

- **CPU**: AMD Ryzen 9 5900XT 3.3 GHz 16-Core Processor [16 Cores, 32 Threads, Base Clock: 3.3GHz]
- **GPU**: Gigabyte GAMING OC GeForce RTX 4060 Ti 16GB [16GB GDDR6 VRAM, Boost Clock: 2.310 GHz]
- **Memory**: Corsair Vengeance LPX 64GB (2 × 32GB) DDR4 [3200MHz Speed, CL16 Latency, Dual Channel Configuration]
- **Storage**: Kingston NV3 NVMe SSD 2TB [PCIe 4.0 ×4 Interface, M.2 2280 Form Factor, Sequential Read: 6000 MB/s]

3.4.2 Software

Due to different code architecture some models are built using TensorFlow [32] while others use PyTorch [33] and as a result have slightly different Software architectures.

- The models that are built on Tensorflow and PyTorch are run on the Linux operating system specifically Ubuntu 22.04, with TensorFlow 2.10 and PyTorch 1.13 to allow GPU acceleration.

Unless stated otherwise all models use Nvidia CUDA toolkit version 11.6 [34].

4 Results

4.1 Results Summary

The bar chart below compares the 3D Average Precision (mAP) of several state-of-the-art object detection models. Among the models evaluated, PointRCNN + VoxelRCNN achieved the highest mAP, followed closely by Complex Yolo v4 and PV-RCNN, indicating strong performance in 3D detection tasks. Fast-point RCNN also performed well, while AVOD and Voxel Net exhibited comparatively lower mAP scores. CaDNN in particular performed poorly with an mAP of around 10%, however this is to be expected of a monocular detector [29]. The results highlight the better performance of voxel-based and point-based deep learning models over traditional approaches in terms of accuracy on 3D object detection.

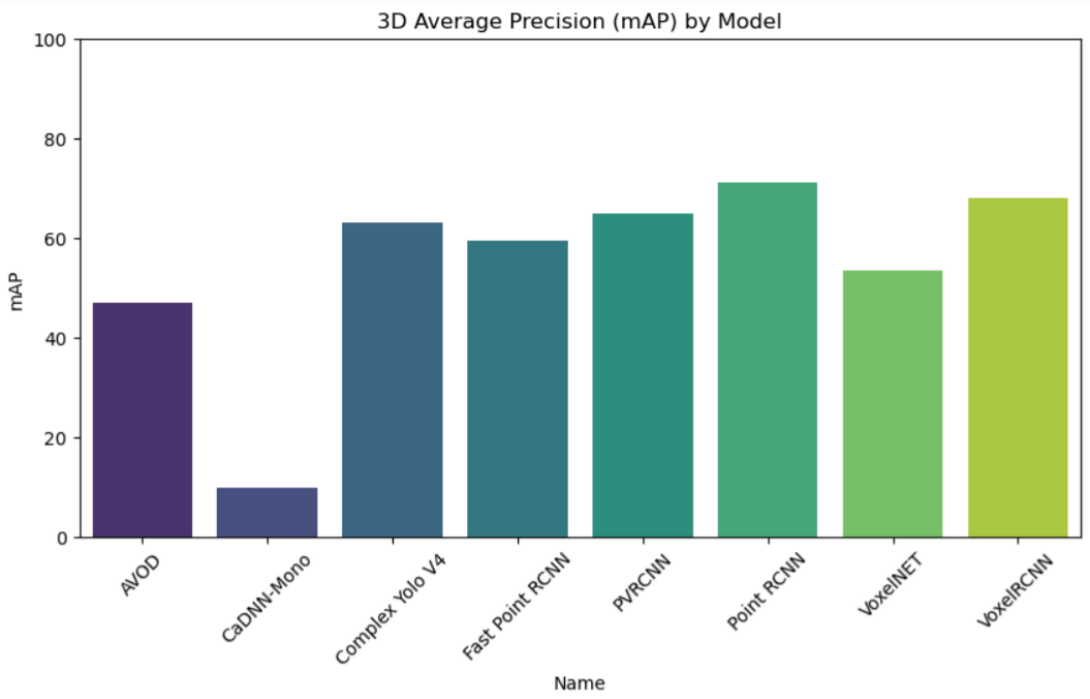

Figure 4: Barchart of model mAP.

We can also see how models compared in mAP for all Cars and for easy difficulties, presented in Table1 and Figure5. While overall mAP provides a comprehensive evaluation, the car-class results (mAP Car) reveal critical architecture-specific behaviours in

vehicle detection. PointRCNN's 81.96% mAP not only leads this category but also explains its strong overall performance (71.17%), as we expect cars to feature heavily in the KITTI dataset, as it was recorded on streets [12]. The 80.05% from Complex Yolo V4 similarly supports its competitive overall mAP (63.25%), while CaDNN-Mono's extremely low 12.55% car mAP directly contributes to its bottom-tier overall performance (9.86%). This apparent correlation tells us that car-class detection capability is often the primary driver of overall scores in automotive 3D detection tasks.
The easy-case results (mAP Easy) show how our models perform under ideal conditions, providing insight into their best performance. PointRCNN's 78.22% demonstrates consistency with its car-class performance, while the gap between VoxelNET's easy-case (59.23%) and car-class (68.27%) mAPs indicates its architecture handles simple cases worse than typical vehicle detection scenarios. Notably, PVRCNN maintains strong easy-case performance (68.89%) despite its modest FPS, suggesting its hybrid design [22] performs well in ideal situations.

Table 2: 3D Object Detection Performance Comparison

Model	**mAP**	**mAP (Car)**	**mAP (Easy)**	**FPS**
AVOD	47.11	67.20	56.36	13.8
CaDNN-Mono	9.86	12.55	12.43	2.9
Complex Yolo V4	63.25	80.05	68.02	31.0
Fast Point RCNN	59.55	77.97	64.22	16.1
PVRCNN	64.89	77.78	68.89	8.8
Point RCNN	71.17	81.96	78.22	10.6
VoxelNET	53.51	68.27	59.23	46.0
VoxelRCNN	68.19	79.48	75.03	22.1

We can see in Figure 5 that processing speeds (FPS) vary dramatically across architectures, creating clear trade-offs between the accuracy metrics discussed previously and practical real world usability. VoxelNET's leading 46.0 FPS comes at the expected cost to accuracy (53.51% overall mAP), while PointRCNN's high accuracy (71.17% overall) came with a computational tradeoff at 10.6 FPS. Complex Yolo V4 appears to be the most balanced contender, delivering 31.0 FPS alongside high accuracy (63.25% overall, 80.05% car). These FPS differences highlight why model selection depends heavily on deployment constraints, as the models with the highest accuracies may not be deployment viable in AVs due to poor FPS. For example, PointRCNN processes 10.6 frames/sec, making it too slow for dense urban or motorway scenes, which can require almost twice the FPS [35].

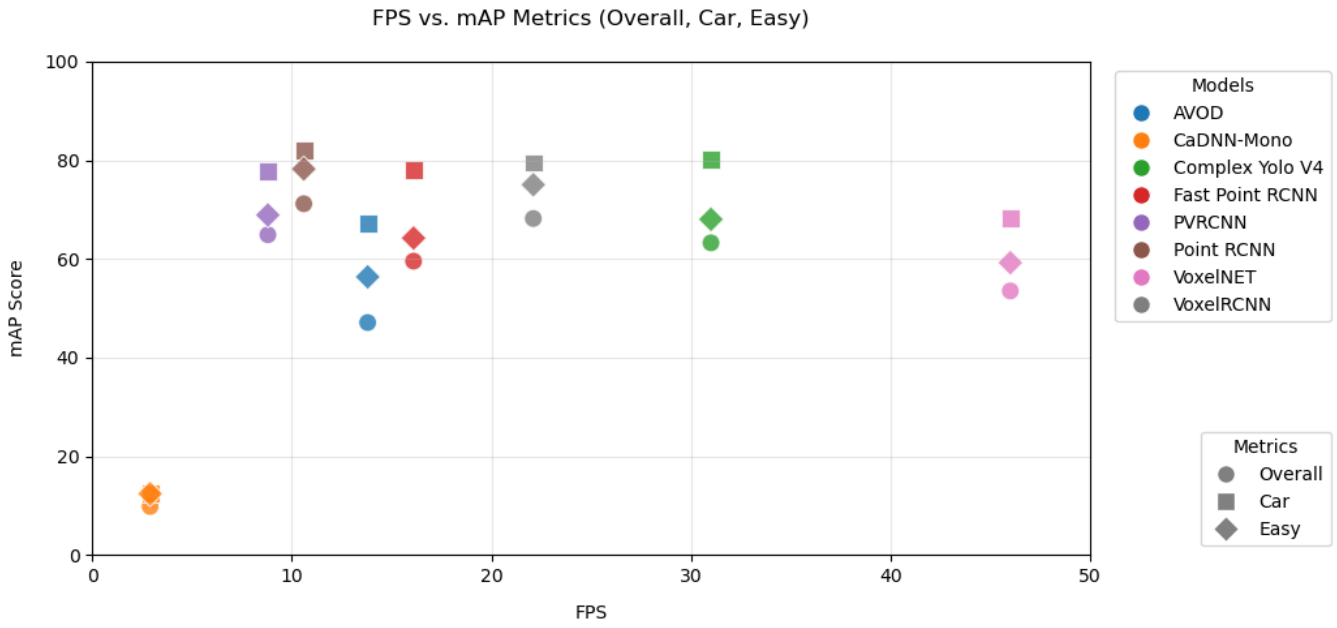

Figure 5: Dotplot of model mAP.

4.2 Point RCNN

This table shows the 3D Average Precision (AP) performance of Point RCNN, across different difficulty levels (Easy/Moderate/Hard) for cars, cyclists, and pedestrians. Higher AP scores indicate better detection accuracy. Cars use a stricter IoU threshold (0.7) due to their larger size, while cyclists and pedestrians use 0.5 to avoid bias [36]. Performance declines with difficulty level, with cars detected most accurately (88.13% Easy) and pedestrians least (51.03% Hard), however it still outperformed most models in terms of mAP as seen in Table 2. We can read each row to see how the model performs for a specific class under varying conditions.

Table 3: Point RCNN Performance Metrics AP(3D)

Model	**Approach**	**Class**	**IOU**	**Easy (%)**	**Moderate (%)**	**Hard (%)**
Point RCNN	Raw Point	car	0.7	88.13	80.71	77.03
Point RCNN	Raw Point	cyclist	0.5	84.31	73.22	70.01
Point RCNN	Raw Point	pedestrian	0.5	62.22	53.89	51.03

4.3 Fast Point RCNN

Fast Point RCNN shows a slight performance drop compared to Point RCNN, particularly in pedestrian detection (45.23% vs. 62.22% AP on easy). However, it still has a competitive accuracy for cars, suggesting a slight trade-off between speed and precision.

Table 4: Fast Point RCNN Performance Metrics AP(3D)

Model	Approach	Class	IOU	Easy (%)	Moderate (%)	Hard (%)
Fast Point RCNN	Raw Point	car	0.7	85.09	77.14	71.67
Fast Point RCNN	Raw Point	cyclist	0.5	62.34	58.25	55.89
Fast Point RCNN	Raw Point	pedestrian	0.5	45.23	41.81	38.51

4.4 Complex Yolo V4

Complex Yolo V4 achieves balanced performance, closely matching Point RCNN in car detection (85.73% AP on easy) but lacking in pedestrian detection. Its cyclist detection remains competitive, though moderate and hard cases see a noticeable decline.

Table 5: Complex Yolo V4 Performance Metrics

Model	Approach	Class	IOU	Easy (%)	Moderate (%)	Hard (%)
Complex Yolo V4	Projection	car	0.7	85.73	77.31	77.12
Complex Yolo V4	Projection	cyclist	0.5	72.37	61.06	59.80
Complex Yolo V4	Projection	pedestrian	0.5	45.97	45.82	44.11

4.5 AVOD

AVOD underperforms relative to raw-point-cloud based models, particularly in pedestrian (35.98% AP on easy) and cyclist detection. Its moderate and hard-case accuracy for all classes is notably weaker, highlighting limitations in complex scenarios.

Table 6: AVOD Performance Metrics

Model	Approach	Class	IOU	Easy (%)	Moderate (%)	Hard (%)
AVOD	Projection	car	0.7	75.91	65.23	60.45
AVOD	Projection	cyclist	0.5	57.18	42.03	36.87
AVOD	Projection	pedestrian	0.5	35.98	26.75	23.60

4.6 VoxelNET

VoxelNET delivers moderate results, with car detection (75.03% AP on easy) trailing Point RCNN but surpassing AVOD. Pedestrian and cyclist performance is inconsistent, with a steep drop in hard-case accuracy, suggesting that it struggles considerable on non-car classes.

Table 7: VoxelNET Performance Metrics

Model	Approach	Class	IOU	Easy (%)	Moderate (%)	Hard (%)
VoxelNET	Voxel	car	0.7	75.03	68.21	61.57
VoxelNET	Voxel	cyclist	0.5	61.14	53.90	46.51
VoxelNET	Voxel	pedestrian	0.5	41.52	38.47	35.27

4.7 VoxelRCNN

VoxelRCNN excels in car detection (87.87% AP on easy), nearly matching Point RCNN, while significantly improving pedestrian recognition over VoxelNET. Cyclist detection remains stable, though hard-case performances dip slightly.

Table 8: VoxelRCNN Performance Metrics

Model	Approach	Class	IOU	Easy (%)	Moderate (%)	Hard (%)
VoxelRCNN	Voxel	car	0.7	87.87	78.27	72.31
VoxelRCNN	Voxel	cyclist	0.5	69.22	65.19	61.83
VoxelRCNN	Voxel	pedestrian	0.5	68.01	57.87	53.12

4.8 PVRCNN

PVRCNN shows strong car detection (81.20% AP on easy) but lags behind VoxelRCNN and Point RCNN. Its pedestrian and cyclist performance is average, but moderate and hard cases reveal room for improvement.

Table 9: PVRCNN Performance Metrics

Model	Approach	Class	IOU	Easy (%)	Moderate (%)	Hard (%)
PVRCNN	Hybrid	car	0.7	81.20	77.26	74.87
PVRCNN	Hybrid	cyclist	0.5	69.81	64.89	61.78
PVRCNN	Hybrid	pedestrian	0.5	55.67	50.30	48.24

4.9 CaDNN

As expected CaDNN-Mono struggles across all classes, with notably low AP scores (14.19% for cars on easy), reflecting the challenges of monocular 3D detection. Its performance is noticeably inferior to other examined methods.

Table 10: CaDNN Performance Metrics

Model	Approach	Class	IOU	Easy (%)	Moderate (%)	Hard (%)
CaDNN-Mono	Hybrid	car	0.7	14.19	12.31	11.16
CaDNN-Mono	Hybrid	cyclist	0.5	11.55	7.84	6.16
CaDNN-Mono	Hybrid	pedestrian	0.5	11.55	7.84	6.16

5 Discussion

5.1 Model Comparisons: Introducing M-Score

The KITTI leaderboards rank models based on mAP, specifically mAP (Moderate). However we know that FPS is just as important as mAP when it comes to real world deployment [37]. In real-world applications such as autonomous driving or robotics FPS is vital, as slow inference can render even the most accurate models unusable in time-sensitive

scenarios.
To address this gap, we propose the M-Score, a composite metric balancing both accuracy and speed:

$$\text{M-Score} = 0.65 \cdot \left(\frac{\text{mAP}}{100}\right) + 0.35 \cdot \left(\frac{\log(\text{FPS}+1)}{\log(61)}\right) \tag{2}$$

This allows us to better compare our models, not just on their accuracy and speed, but also on their realworld usability. Most scores will lie between 0 and 1 with higher scores indicating better suitability to real world use in AVs.:

Table 11: Model Performance with M-Score

Model Name	**mAP**	**FPS**	**M-Score**
VoxelRCNN	68.187778	22.1	0.711
Complex Yolo V4	63.254444	31.0	0.706
VoxelNET	53.513333	46.0	0.676
Point RCNN	71.172222	10.6	0.671
Fast Point RCNN	59.547778	16.1	0.629
PVRCNN	64.891111	8.8	0.616
AVOD	47.111111	13.8	0.536
CaDNN-Mono	9.862222	2.9	0.180

5.2 Summary and Interpretation of Key Results

5.2.1 Summary

In this paper 8 different state -of-the-art models were tested on the KITTI dataset. Out of the models tested Point RCNN achieved the highest accuracy (mAP 71.17) but is slow (10.6 FPS). VoxelRCNN better balances accuracy (mAP 68.19) and speed (22.1 FPS), while Complex Yolo V4 trades slightly lower accuracy (mAP 63.25) for better speed (31.0 FPS). VoxelNET is the fastest (46.0 FPS) but less accurate (mAP 53.51). CaDNN-Mono performs poorly in both accuracy (mAP 9.86) and speed (2.9 FPS).
For overall performance (M-Score), VoxelRCNN (0.711) and Complex Yolo V4 (0.706) rank highest, combining accuracy and speed effectively. Point RCNN (0.671) suffers in M-Score due to low FPS, while CaDNN-Mono (0.180) scores worst.
The results (Figure 5 demonstrate a clear trade-off between detection accuracy (mAP) and inference speed (FPS) across different 3D object detection models. High-accuracy models like Point RCNN (71.17 mAP) achieve superior detection performance but suffer from slower inference speeds (10.6 FPS), making them less suitable for real-time applications. Conversely, faster models like VoxelNET (46 FPS) offer rapid processing but at the cost of reduced accuracy (53.51 mAP). This inverse relationship highlights one of the largest challenges in 3D object detection: balancing precision with computational efficiency [7] and highlights the need for application-specific model selection.
Application-specific model selection could exist in the form of an AV using a different object detection system depending on the working environment, with high-precision models (lower FPS) being used for everyday urban driving (such as dense city traffic or pedestrian-heavy zones) and high-FPS (lower precision) models being used for high-speed environments (such as highway driving or open rural roads). A similar application specific model selection is already commonly used by drones and UAVs in industry [38].

5.3 Implication of Key Results

Now we examine our results in terms of our research questions:

> *How do state-of-the-art deep learning techniques compare in accuracy, efficiency, and speed for LiDAR-based 3D object detection in autonomous driving scenarios?*

In addressing our first research question, the findings reveal that state-of-the-art deep learning techniques for LiDAR-based 3D object detection vary significantly in accuracy, efficiency, and speed. The raw-point-based methods [6; 7] examined excel in accuracy but performed worse in terms of speed, while voxel-based approaches [23; 28] offer better speed-accuracy trade-offs. Notably, Complex Yolo V4 demonstrates that optimized architectures can achieve competitive accuracy without sacrificing real-time performance, making it a strong candidate for autonomous driving scenarios, despite not having the highest mAP or FPS scores.

> *How do hybrid LiDAR-based 3D object detection approaches compare to pure projection-based, voxel-based, and raw-point-based methods in terms of accuracy and computational efficiency?*

In terms of our second question: Hybrid methods appeared to struggle compared to more traditional methods, likely due to their additional computational overhead. Our analysis reveals hybrid LiDAR methods like PVRCNN (64.89 mAP, 8.8 FPS) face efficiency trade-offs, struggling to match the speed of optimized voxel-based approaches such as VoxelRCNN (68.19 mAP, 22.1 FPS). While hybrids aim to combine multiple paradigms' strengths, their computational overhead often outweighs benefits for real-time applications. The weak performance of CaDNN-Mono (9.86 mAP, 2.9 FPS) is expected of Monocular models [29] but may not be a fair representation of all hybrid models. Whereas in comparison, pure methods show more consistent performance patterns. Voxel-based techniques offer the best accuracy-speed balance, while point-based models prioritize precision. These findings suggest pure methods remain preferable for most autonomous driving tasks, unless specific detection challenges truly require hybrid architectures' multimodal processing capabilities. Future hybrid models must significantly improve computational efficiency to compete with state-of-the-art pure approaches.

6 Conclusion

6.1 Limitations and Future Work

Although this project aims to provide a comprehensive analysis and comparison of 3D detection algorithms, it is important to note that the evaluation is based on benchmark datasets and simulated performance. The lack of real-world testing limits insight into how AVs that deploy this technology would act in the real world. This is important to note for any deployment in actual AV systems.
The KITTI dataset has received criticism [39] as it only features good weather during daytime, meaning that models may may not to perform as well in real world testing, or on datasets featuring adverse weather conditions (such as the Oxford RoboCar Dataset [40] which heavily features rain and night scenes). However, we believe that it is still well

suited for the purpose of model comparison.
One weakness of this project is the inclusion of monocular methods (e.g. CaDNN - Mono[29]) under the broad label of hybrid models. If this project were to be repeated Monocular methods would be treated as their own category, as they often feature very low mAP[41], to prevent them from negatively skewing the hybrid category as a whole.
Due to the time constraints of this masters project being only 12 weeks we were only able to compare 8 different models on the KITTI dataset and on our hardware.
Future work would to be extend this project to more models creating a freely accessible database of model performances on the same architecture. This would enable more fair and comprehensive benchmarking and better enable researchers and practitioners in the field to make comparisons between models. Future work could also feature the proposed M-Score metric to allow future researchers to instantly compare models at a glance.
As mentioned in *Acknowledging Limitations*, future work could view hybrid approaches and monocular approaches as seperate categories for better comparisons.

References

[1] S. S. Rampalli, P. Mehta, P. Vyas, Shashwat, and J. Dauwels, "Redesigning infrastructure for autonomous vehicles and evaluating its impact on traffic," in *2020 Forum on Integrated and Sustainable Transportation Systems (FISTS)*, 2020, pp. 242–246.

[2] T. Winkle, *Safety Benefits of Automated Vehicles: Extended Findings from Accident Research for Development, Validation and Testing*, 05 2016, pp. 335–364.

[3] D. Kumari and S. Bhat, "Application of artificial intelligence technology in tesla-a case study," *International Journal of Applied Engineering and Management Letters (IJAEML)*, vol. 5, no. 2, pp. 205–218, 2021.

[4] F. Lin, H. Yan, J. Li, Z. Liu, L. Lu, Z. Ba, and K. Ren, "Phade: Practical phantom spoofing attack detection for autonomous vehicles," *IEEE Transactions on Information Forensics and Security*, vol. PP, pp. 1–1, 01 2024.

[5] M. Caro, H. Tabani, J. Abella, F. Moll, E. Morancho, R. Canal, J. Altet, A. Calomarde, F. J. Cazorla, A. Rubio, P. Fontova, and J. Fornt, "An automotive case study on the limits of approximation for object detection," *Journal of Systems Architecture*, vol. 138, p. 102872, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1383762123000516

[6] S. Shi, X. Wang, and H. Li, "Pointrcnn: 3d object proposal generation and detection from point cloud," 2019. [Online]. Available: https://arxiv.org/abs/1812.04244

[7] Y. Chen, S. Liu, X. Shen, and J. Jia, "Fast point r-cnn," 2019. [Online]. Available: https://arxiv.org/abs/1908.02990

[8] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *International Journal of Robotics Research (IJRR)*, 2013.

[9] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, V. Vasudevan, W. Han, J. Ngiam, H. Zhao, A. Timofeev, S. Ettinger, M. Krivokon, A. Gao, A. Joshi, Y. Zhang, J. Shlens, Z. Chen,

and D. Anguelov, "Scalability in perception for autonomous driving: Waymo open dataset," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[10] X. Huang, P. Wang, X. Cheng, D. Zhou, Q. Geng, and R. Yang, "The apolloscape open dataset for autonomous driving and its application," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 10, p. 2702–2719, Oct. 2020. [Online]. Available: http://dx.doi.org/10.1109/TPAMI.2019.2926463

[11] K. T. e. a. H. Caesar, J. Kabzan, "Nuplan: A closed-loop ml-based planning benchmark for autonomous vehicles," in *CVPR ADP3 workshop*, 2021.

[12] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[13] Y. Li, L. Ma, Z. Zhong, F. Liu, M. Chapman, D. Cao, and J. Li, "Deep learning for lidar point clouds in autonomous driving: A review," *IEEE Transactions on Neural Networks and Learning Systems*, vol. PP, pp. 1–21, 08 2020.

[14] Y. Wang, S. Wang, Y. Li, and M. Liu, "A comprehensive review of 3d object detection in autonomous driving: Technological advances and future directions," 2024. [Online]. Available: https://arxiv.org/abs/2408.16530

[15] Y. Ma, Z. Wang, H. Yang, and L. Yang, "Artificial intelligence applications in the development of autonomous vehicles: a survey," *IEEE/CAA Journal of Automatica Sinica*, vol. 7, no. 2, pp. 315–329, 2020.

[16] M. Simon, S. Milz, K. Amende, and H.-M. Gross, "Complex-yolo: Real-time 3d object detection on point clouds," 2018. [Online]. Available: https://arxiv.org/abs/1803.06199

[17] J.-q. Luo, Y. Shi, and C. Xie, "Fsspd: Fast single stage pedestrian detector for autonomous driving," *DEStech Transactions on Environment, Energy and Earth Sciences*, 02 2019.

[18] L. Jiao, F. Zhang, F. Liu, S. Yang, L. Li, Z. Feng, and R. Qu, "A survey of deep learning-based object detection," *IEEE Access*, vol. 7, pp. 128 837–128 868, 2019.

[19] S. Y. Alaba and J. E. Ball, "A survey on deep-learning-based lidar 3d object detection for autonomous driving," *Sensors*, vol. 22, no. 24, 2022. [Online]. Available: https://www.mdpi.com/1424-8220/22/24/9577

[20] A. Pravallika, M. F. Hashmi, and A. Gupta, "Deep learning frontiers in 3d object detection: A comprehensive review for autonomous driving," *IEEE Access*, vol. 12, pp. 173 936–173 980, 2024.

[21] Y. Wu, Y. Wang, S. Zhang, and H. Ogai, "Deep 3d object detection networks using lidar data: A review," *IEEE Sensors Journal*, vol. 21, no. 2, pp. 1152–1171, 2021.

[22] S. Shi, C. Guo, L. Jiang, Z. Wang, J. Shi, X. Wang, and H. Li, "Pv-rcnn: Point-voxel feature set abstraction for 3d object detection," 2021. [Online]. Available: https://arxiv.org/abs/1912.13192

[23] Y. Zhou and O. Tuzel, "Voxelnet: End-to-end learning for point cloud based 3d object detection," 2017. [Online]. Available: https://arxiv.org/abs/1711.06396

[24] L. Chen, G. Li, W. Xie, J. Tan, Y. Li, J. Pu, L. Chen, D. Gan, and W. Shi, "A survey of computer vision detection, visual slam algorithms, and their applications in energy-efficient autonomous systems," *Energies*, vol. 17, no. 20, 2024. [Online]. Available: https://www.mdpi.com/1996-1073/17/20/5177

[25] F. Tarsha-Kurdi, T. Landes, P. Grussenmeyer, and M. Koehl, "Model-driven and data-driven approaches using lidar data: Analysis and comparison," *PIA 2007 - Photogrammetric Image Analysis, Proceedings*, 01 2007.

[26] M. Ye, S. Xu, and T. Cao, "Hvnet: Hybrid voxel network for lidar based 3d object detection," 06 2020, pp. 1628–1637.

[27] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander, "Joint 3d proposal generation and object detection from view aggregation," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE Press, 2018, p. 1–8. [Online]. Available: https://doi.org/10.1109/IROS.2018.8594049

[28] J. Deng, S. Shi, P. Li, W. Zhou, Y. Zhang, and H. Li, "Voxel r-cnn: Towards high performance voxel-based 3d object detection," 2021. [Online]. Available: https://arxiv.org/abs/2012.15712

[29] C. Reading, A. Harakeh, J. Chae, and S. L. Waslander, "Categorical depth distribution network for monocular 3d object detection," 2021. [Online]. Available: https://arxiv.org/abs/2103.01100

[30] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds," 2019. [Online]. Available: https://arxiv.org/abs/1812.05784

[31] O. D. Team, "Openpcdet: An open-source toolbox for 3d object detection from point clouds," https://github.com/open-mmlab/OpenPCDet, 2020.

[32] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[33] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," 2019. [Online]. Available: https://arxiv.org/abs/1912.01703

[34] NVIDIA, P. Vingelmann, and F. H. Fitzek, "Cuda, release: 10.2.89," 2020. [Online]. Available: https://developer.nvidia.com/cuda-toolkit

[35] W. E. Villegas, S. Sánchez-Viteri, and S. Luján-Mora, "Real-time recognition and tracking in urban spaces through deep learning: A case study," *IEEE Access*, vol. 12, pp. 95 599–95 612, 2024.

[36] D. Katare, D. S. Noguero, S. Park, N. Kourtellis, M. Janssen, and A. Y. Ding, "Analyzing and mitigating bias for vulnerable road users by addressing class imbalance in datasets," *IEEE Open Journal of Intelligent Transportation Systems*, vol. 6, pp. 590–604, 2025.

[37] R. Jenkin and P. Kane, "Fundamental imaging system analysis for autonomous vehicles," *Electronic Imaging*, vol. 30, no. 17, pp. 105–1–105–1, 2018. [Online]. Available: https://library.imaging.org/ei/articles/30/17/art00002

[38] A. Phadke and F. A. Medrano, "Examining application-specific resiliency implementations in uav swarm scenarios," *Intelligence Robotics*, vol. 3, no. 3, 2023. [Online]. Available: https://www.oaepublish.com/articles/ir.2023.27

[39] M. J. Mirza, C. Buerkle, J. Jarquin, M. Opitz, F. Oboril, K.-U. Scholl, and H. Bischof, "Robustness of object detectors in degrading weather conditions," 2021. [Online]. Available: https://arxiv.org/abs/2106.08795

[40] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, "1 Year, 1000km: The Oxford RobotCar Dataset," *The International Journal of Robotics Research (IJRR)*, vol. 36, no. 1, pp. 3–15, 2017. [Online]. Available: http://dx.doi.org/10.1177/0278364916679498

[41] X. Weng and K. Kitani, "Monocular 3d object detection with pseudo-lidar point cloud," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, Oct 2019.