UTILIZING ARTIFICIAL NEURAL NETWORKS FOR 3D OBJECT DETECTION IN TRAFFIC VEHICLES TO SUPPORT AUTONOMOUS VEHICLE TECHNOLOGY DEVELOPMENT

Farid Rahman Nurdin[1)]

[1)] *Program Studi Informatika, Fakultas Teknik, Universitas Muhammadiyah Surabaya*
Jl Sutorejo No. 59, Surabaya
Email : farid.rahman.nurdin-2021@ft.um-surabaya.ac.id[1)]

Abstrak

Peningkatan jumlah kendaraan berkontribusi terhadap tingginya angka kecelakaan lalu lintas, yang sebagian besar disebabkan oleh kesalahan manusia. Sebagai contoh, pada tahun 2024, jumlah mobil mencapai 1.475 miliar, dan sekitar 61% kecelakaan disebabkan oleh human error yang berpotensi berujung pada kematian. Penggunaan kendaraan otonom dapat menjadi solusi dari permasalahan tersebut guna mengurangi kontak kendali secara langsung oleh manusia. Pada perkembangannya, penggunaan metode deteksi objek 3D kendaraan lalu lintas berdasarkan sensor kamera monokuler dan sensor Light Detection and Ranging (LiDAR) yang diolah melalui algoritma YOLOv3 dengan backbone Darknet-53, masih sanggup menjadi opsi untuk pengolahan dataset citra. Hasil dari pengolahan dataset tersebut yaitu 0,60 Mean Average Precision (mAP) dengan threshold Intersection of Union (IoU) sebesar 0,5. Sehingga potensi metode deteksi objek 3D kendaraan lalu lintas pada kendaraan otonom dapat dikembangkan menjadi lebih akurat dan presisi dengan penerapan algoritma terbarukan dan model objek yang lebih lengkap di dalam dataset.

Kata kunci: *Deteksi objek 3D, kamera monokuler, kendaraan otonom, LiDAR, YOLOv3.*

Abstract

The increasing number of vehicles contributes to many traffic accidents, most of which are caused by human error. For example, in 2024, the number of cars will reach 1.475 billion, and around 61% of accidents are caused by human error, which has the potential to lead to death. The use of autonomous vehicles can be a solution to this problem to reduce direct human control contact. In its development, the use of a 3D traffic vehicle object detection method based on a monocular camera sensor and a Light Detection and Ranging (LiDAR) sensor processed through the YOLOv3 algorithm with a Darknet-53 backbone is still an option for image dataset processing. The results of the dataset processing are 0.60 Mean Average Precision (mAP) with an Intersection of Union (IoU) threshold of 0.5. So, the potential for the 3D traffic vehicle object detection method in autonomous vehicles can be developed to be more accurate and precise by applying the latest algorithms and more complete object models in the dataset.

Keywords : *3D object detection, autonomous vehicle, LiDAR, monocular camera, YOLOv3*

1. Introduction

The advancement of mobility in human civilization is accompanied by technological advances in vehicles [1]. Although it has a positive impact because it helps humans to move quickly, in reality, excessive vehicle production also has a series of negative impacts. This is evidenced by the world's vehicle population, which is dominated by cars, reaching 1475 billion in 2024 [2]. The large population figure is correlated with the level of traffic accidents that can lead to death. For example, the following is a statistical picture of the death rate due to traffic accidents in Indonesia when compared to other countries in Figure 1, according to the World Health Organization (WHO) [3]. Most of the causes of these accidents are human error itself (human error), as much as 61% according to the Traffic Corps (Korlantas Polri) [4].

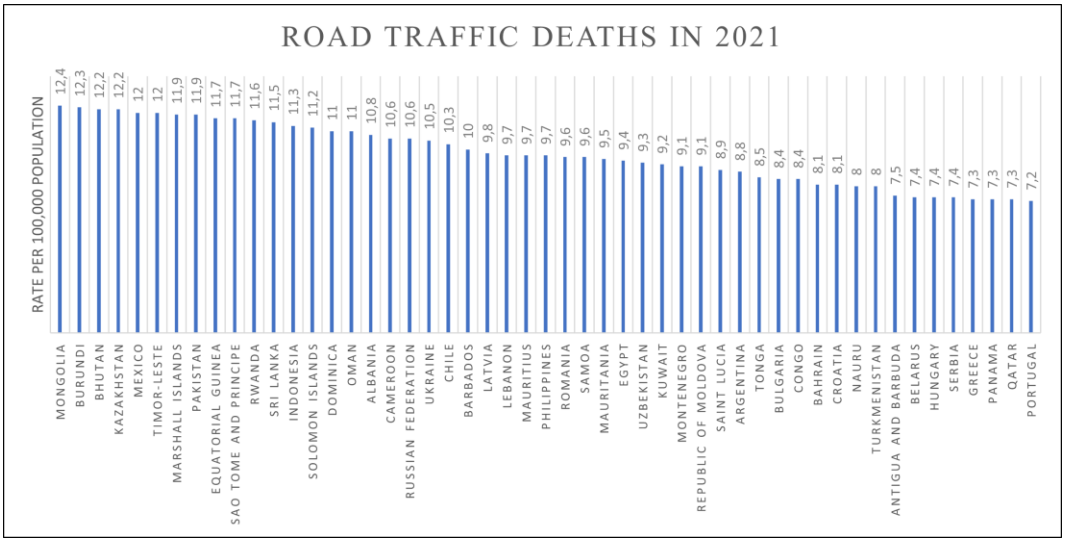

Figure 1. Number of Road Accident Deaths by Country in 2021

The impact of traffic accidents due to human error is indeed unavoidable. The causes are influenced by several factors such as fatigue, lack of concentration while driving or poor decision making [5]. So, in reducing accident factors, an example of a solution in terms of vehicle technology development is changing driving practices from Human-driven vehicles (HDVs) to Self-driving vehicles (SDVs) [6].

The comparison of the two in terms of safety is that SDVs must be four to five times safer than HDVs [6]. In addition, SDVs still require human supervision, while Autonomous Vehicles (AVs) are designed to no longer require human intervention [7]. So the ability of sensors and algorithms used to process data needs to be taken into account in optimizing detection capabilities and maintaining user safety [8]. Especially object detection according to its three-dimensional (3D) shape, which is more relevant to the depiction of the surrounding environment.

This study aims to improve 3D object detection accuracy and efficiency in challenging conditions to reduce computational burdens. Mitigate traffic fatalities by replacing error-prone human drivers with AI-driven perception systems. And advocate for infrastructure and regulatory frameworks that support the integration of autonomous vehicles into existing transportation ecosystems.

2. Basic Theory

Here are some fundamental theories related to the research being conducted.

2.1. Autonomous Vehicles

Autonomous vehicles are self-operating systems that rely on sensor technologies, artificial intelligence, and software to control and navigate without human involvement [9]. According to the Society of Automotive Engineers (SAE), these vehicles are classified into five levels of autonomy [7]. Level 1 enables partial functions like speed control, Level 2 allows for automated driving with driver oversight, and Level 3 supports autonomous operation under specific conditions. Level 4 vehicles operate independently in certain areas, while Level 5 vehicles are fully autonomous with no driver input required [7], [10].

To navigate complex environments, autonomous vehicles use a range of sensors, including LIDAR, radar, stereo cameras, and ultrasonic sensors [8]. LIDAR is particularly important for creating detailed 3D maps of the surroundings, enabling accurate object detection, especially in higher-level autonomous systems (Level 4 and 5) [11]. This sensor fusion allows vehicles to detect other cars, obstacles, and barriers, helping them make autonomous decisions, avoid collisions, and maintain safety and efficiency in traffic.

2.2. Types of Object Classes to be Detected

In 3D object detection research for traffic environments, accurately identifying surrounding objects is vital for enhancing the safety and efficiency of autonomous vehicles [12]. A widely used approach involves categorizing objects into three main classes—Cars, Cyclists, and Pedestrians—based on the KITTI dataset [13]. Each class has unique characteristics in terms of size, shape, and movement, influencing detection strategies. Cars are prioritized due to their frequent presence and stable form, which makes them easier to detect using sensors like monocular cameras and LiDAR 016[14]. Cyclists, being smaller and more dynamic, require precise motion prediction and rapid sensor response [14]. Pedestrians, with unpredictable behavior and variable appearance, must be detected early to ensure vehicle safety, relying heavily on detailed LiDAR data for accurate localization [14].

However, the YOLOv3 model used in this research is trained only on these three KITTI dataset classes, limiting its ability to recognize objects beyond them. Accurate object labelling, including 2D and 3D bounding boxes and geometric attributes like orientation and size, is critical to help the model distinguish between these categories effectively [15]. This precision enables the autonomous system to better understand the surrounding environment, ensuring accurate, timely, and safe decisions in complex traffic scenarios [8].

2.3. Comparison of 3D Object Detection and Classification Sensors

Other research [16] conducted a comparative study on the performance of various sensors for 3D object detection, highlighting the effectiveness of both radar and LiDAR as shown in Figure 2. While FIR (Far Infrared) sensors, commonly known as thermal cameras, are useful in low-light conditions due to their ability to detect heat-based radiation, they struggle with accuracy in uniform temperatures and extreme weather. Among the compared sensors, LiDAR stands out as the most capable for visualizing 3D maps due to its high spatial accuracy. However, its performance declines in adverse conditions like rain or fog, which makes sensor fusion a more robust solution.

Performance Criterion	RADAR	LiDAR	RGB Camera
Object detection	Good	Good	Fair
Object classification	Poor	Fair	Good
Distance inference	Good	Good	Fair
Detecting edge	Poor	Good	Good
Visibility range	Good	Fair	Fair
Adverse weather performance	Good	Fair	Poor
Performance in low light condition	Good	Good	Fair

Figure 2. Performance Comparison of 3D Object Detection Sensors and Devices [16]

To overcome individual sensor limitations, combining LiDAR with a monocular camera is considered optimal for 3D object visualization in this study. Both sensors utilize light-based technology and complement each other—LiDAR offers precise spatial data, while the monocular camera provides rich visual information for object recognition. This fusion improves detection performance, especially in challenging environments or low-light situations, where LiDAR compensates for the monocular camera's limitations, ensuring more reliable 3D perception in autonomous vehicle systems.

2.4. 3D Object Detection Algorithm in Traffic Vehicles

3D object detection in traffic vehicles often relies on image data from monocular cameras, making it closely tied to image processing [17]. This study uses YOLOv3 due to its simple, flexible architecture, low computational requirements [18], and strong community support, despite newer versions like YOLOv10 and YOLOv11 offering improved features. YOLOv3, built on the Darknet-53 backbone, performs well in multi-scale detection and is effective with diverse traffic objects in the KITTI dataset [19]. Its widespread use, availability of pre-trained models, and proven stability make it a practical and efficient choice for object detection in autonomous vehicle systems [20].

3. Methodologyy

3.1. Data Collection

The data collection process uses a traffic vehicle dataset, specifically the KITTI dataset, readily accessible online. This dataset meets the requirements for four sub-datasets: calibration, images, labels, and velodyne. The most suitable KITTI dataset version for the YOLOv3 algorithm is the 3D Object Detection Evaluation 2017 [21]. Table 1 presents details on the quantity of KITTI data used according to the needs of the YOLOv3 model.

Table 1. KITTI 3D Object Detection Evaluation 2017 Dataset Criteria for YOLOv3

No	Sub Dataset	Number of Training Datasets	Number of Testing Datasets
1	Calibration	7481	7518
2	Image	7481	7518
3	Label	7481	0
4	Velodyne	7481	7518

3.2. Data Processing

The image sub-dataset from KITTI has a consistent format of 1242 × 375 pixels, with 96 dpi resolution and 24-bit depth. Object labelling is based on three predefined classes in the label sub-dataset, simplifying data processing by allowing easy adjustment of dataset folder locations to fit algorithmic needs. The YOLOv3 algorithm is implemented using Python, requiring a Python Virtual Environment to manage dependencies, which can be operated through a Command Line Interface like CMD or a text editor such as Visual Studio Code.

3.3. Research Flow Design and Planning

This process includes several key components essential to the research, such as: the research method flowchart shown in Figure 3, the YOLOv3 architecture illustrated in Figure 4, and the research design structured around the YOLOv3 architecture presented in Figure 5.

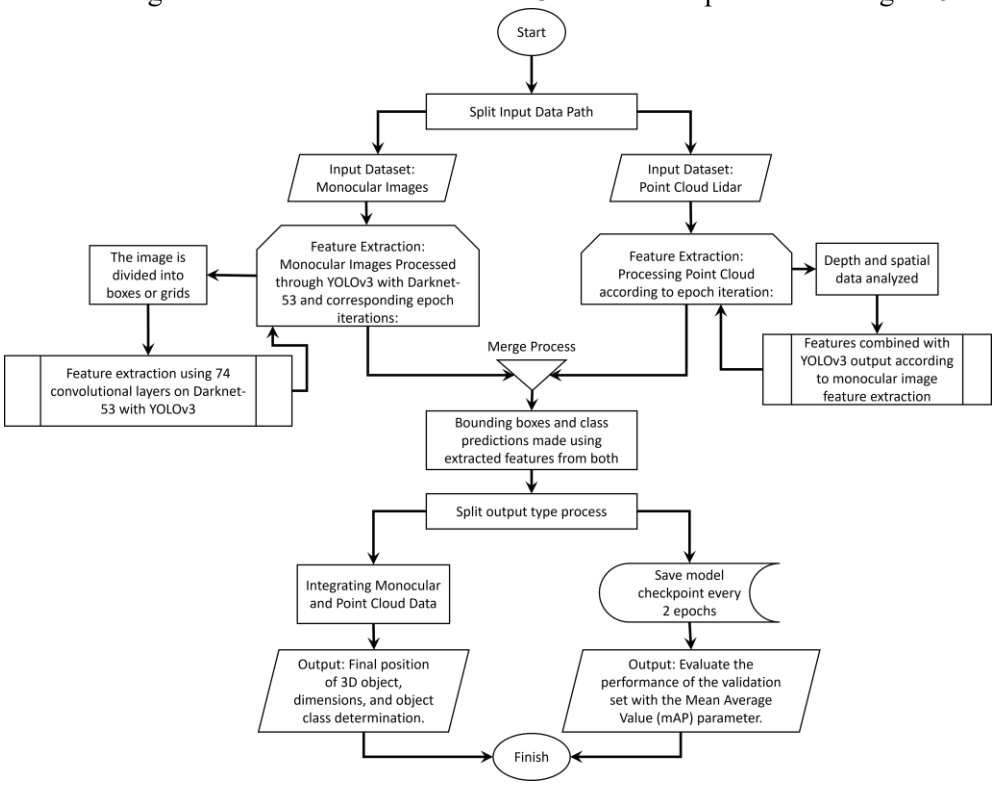

Figure 3. Flowchart Design of Method

This study employs the YOLOv3 architecture, which uses the Darknet-53 backbone [22] consisting of 74 convolutional layers built with residual blocks and skip connections to extract features at multiple image scales. As a one-stage object detection model [23], YOLOv3 emphasizes efficiency and speed while improving accuracy, especially for complex images and small objects, through a multi-scale prediction approach. The model operates within the Darknet framework, an open-source deep learning platform optimized for image processing. Figure 4 illustrates the YOLOv3 architecture with its Darknet-53 backbone.

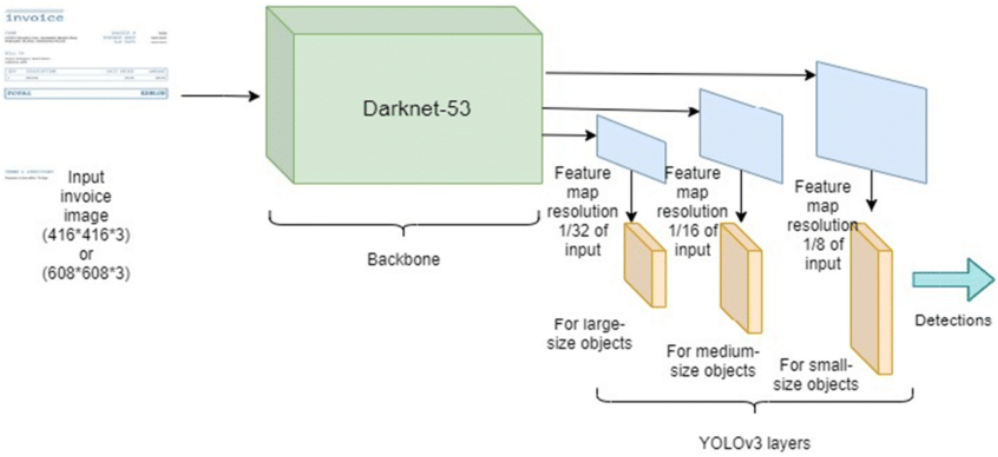

Figure 4. YOLOv3 architecture illustrated [24]

Figure 5 presents the research design adapted from the YOLOv3 architecture, starting with an input stage that applies a fusion method to integrate data from monocular cameras and LiDAR sensors. At the output stage, the system generates 3D bounding boxes around detected objects, visualized through monocular (2D) images from the Front Camera View and 3D point cloud maps using both Front Camera and Bird's Eye View (BEV) perspectives.

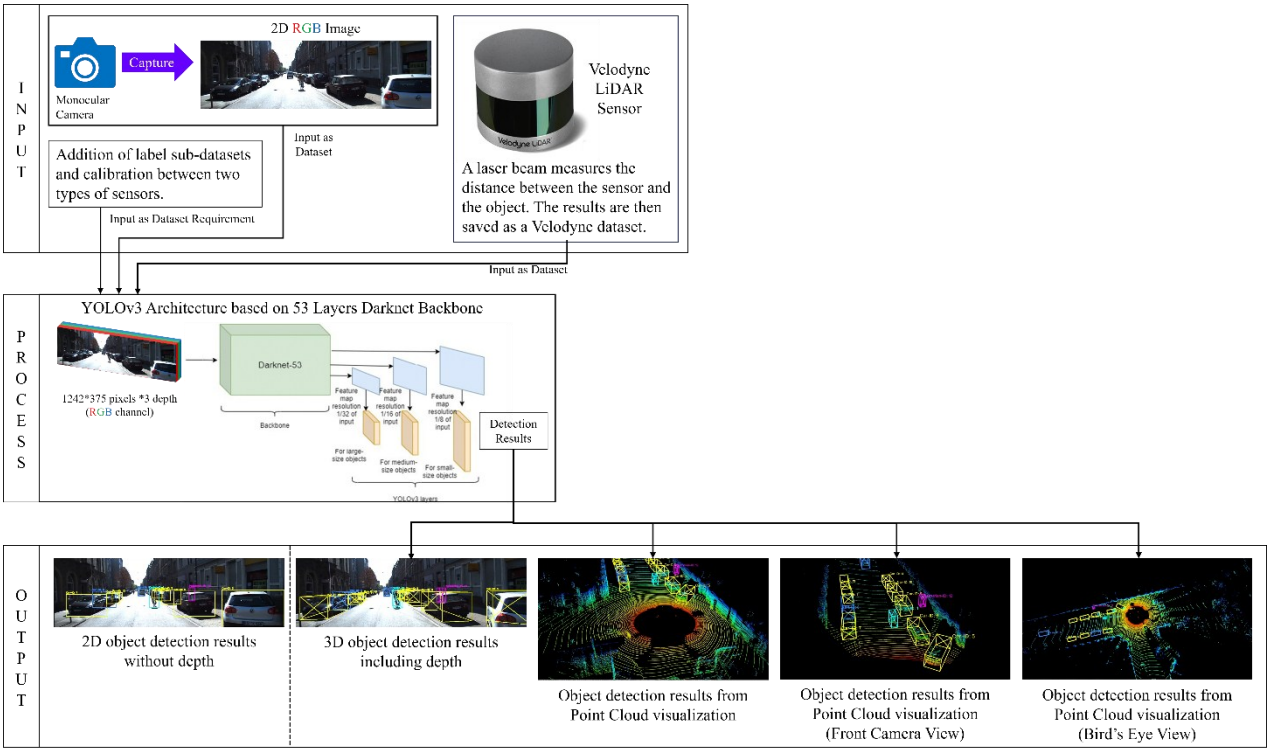

Figure 5. Design of 3D Object Detection for Traffic Vehicles Referencing the YOLOv3 Architecture

3.4. Training Planning

The training process for 3D object detection involves several key components: backpropagation, loss function, and optimization methods.

- Backpropagation is a learning algorithm that efficiently updates neural network weights. It consists of two stages: the forward pass, where input images are processed to predict bounding boxes, object classes, and confidence scores, and the backwards pass, where the difference between predictions and actual values is used to calculate errors and update weights through gradients.
- The loss function measures how well the model performs. It includes three parts: localization loss (error in bounding box prediction), confidence loss (error in object presence prediction), and class probability loss (error in classifying objects). Each of these components helps guide model improvements during training.
- Optimization adjusts network weights to minimize the loss function using algorithms like Stochastic Gradient Descent (SGD), Momentum, RMSProp, and Adam. These methods use gradients from backpropagation to refine the model, with Adam offering adaptive learning rates by combining the strengths of Adagrad and RMSProp for more efficient training in YOLO-based systems.

3.5. Evaluation Plan

In this study, the fusion method is applied by combining two types of input datasets, 2D images and 3D LiDAR maps, to enhance object detection performance. The evaluation is conducted using two primary metrics: the confusion matrix and 3D Mean Average Precision (mAP) [25]. Table 2 presents the confusion matrix, which evaluates model performance through key indicators such as True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN). This table reflects the model's prediction accuracy by summarizing correct and incorrect detections of 3D objects in the test data.

Table 2. KITTI 3D Object Detection Evaluation 2017 Dataset Criteria for YOLOv3

Confusion Matrix		Predicted Class	
		TRUE	FALSE
Observed	TRUE	True Positive or TP: indicates the model is correct in classifying the number of positive data.	False Positive or FP: indicates the model is wrong in classifying the number of positive data.
	FALSE	False Negative or FN: indicates the model is wrong in classifying the amount of negative data.	True Negative or TN: indicates the model is correct in classifying the amount of negative data.

The 3D Mean Average Precision (mAP) is a key metric used to evaluate the performance of 3D object detection models. It is based on the Intersection over Union (IoU), which assesses whether a predicted object correctly matches the ground truth. mAP is calculated by averaging precision scores across different IoU thresholds, such as IoU ≥ 0.5 or ≥ 0.7. Higher IoU values reflect more accurate predictions, leading to a higher mAP score, indicating the model's overall effectiveness.

Both evaluation metrics confusion matrix and 3D mAP play complementary roles in assessing the performance of the YOLOv3 algorithm in this study. The confusion matrix offers a general view of prediction accuracy, while the 3D mAP highlights the model's precision in detecting objects in three-dimensional space. Together, they provide a comprehensive evaluation of YOLOv3's effectiveness in identifying 3D traffic objects, forming the foundation for conclusions about the method's impact on autonomous vehicle technology development.

4. Results and Discussion

4.1 Description of the KITTI 3D Object Dataset in Traffic Used

The KITTI dataset was used in this study, comprising two types of data: monocular images and LiDAR point clouds. It contains a total of 7481 training samples and 7518 testing samples, with objects annotated into three categories: cars, pedestrians, and cyclists, resulting in 80,256 labelled objects. The monocular images have a resolution of 1242 × 375 pixels and a 24-bit colour depth.

For model training, 6000 samples from the training set were used with labels and annotations, while 1481 samples were reserved for validation. Each object category was assessed using a custom mean Average Precision (mAP) script, employing an Intersection over Union (IoU) threshold of ≥ 0.5 during the evaluation phase.

This IoU threshold was selected because it is widely accepted as a standard, providing a balance between sufficient accuracy and computational efficiency, especially relevant when working with limited datasets like KITTI [26]. Although an IoU ≥ 0.7 offers stricter evaluation criteria and can indicate higher-quality detections, it may unfairly lower performance scores in practical scenarios [26]. Therefore, this study adopts an IoU ≥ 0.5 as a reasonable benchmark for early-stage evaluation, with the potential to raise the threshold to IoU ≥ 0.7 in later stages once the model has demonstrated stability.

4.2 Python Virtual Environment Setup and Its Modules

Before processing the training dataset, a Python virtual environment named yolo3_3d must be configured to ensure the YOLOv3 algorithm can optimally handle the KITTI dataset. This environment is tailored with the necessary modules to support the training process effectively.

Table 3. Modules Requirement

Module Name	Version
Python	3.12.4
Pytorch-cuda	12.1
Cuda-runtime	12.1.0=0
Torch	2.3.1
Torchaudio	2.3.1
Torchvision	0.18.1

The training process utilizes an NVIDIA GeForce RTX 4050 Laptop GPU with 6 GB of VRAM, meaning the training outcomes are also influenced by the hardware. This is because GPU VRAM plays a crucial role in handling the computational load during training.

4.3 3D Object Dataset Training Results Evaluation Table

Before training the dataset, several parameters in the Python code must be configured, including batch size, number of epochs, anchor settings, and learning rate. These settings influence how efficiently and accurately the YOLOv3 algorithm can process the KITTI dataset.

The batch size determines how many data samples are processed in one iteration. Larger batches improve GPU efficiency but use more memory, while smaller batches allow more frequent updates but can lead to unstable training. Epochs refer to complete passes through the training data. Multiple epochs help the model learn better, but too few can cause underfitting, and too many can lead to overfitting.

Anchors for 3D objects define the expected size, shape, and orientation of the bounding boxes, helping the model better detect objects with varying dimensions in 3D space. Meanwhile, the learning rate controls the step size for updating model weights; a value too high can destabilize training, while a value too low can slow it down or result in poor solutions.

During training, both the number of epochs and batch size are adjusted to optimize performance. After several epochs, the model creates checkpoints to save progress and allow evaluation. These checkpoints help track improvements and assess mAP (mean Average Precision) results over time. Out of the 7481 dataset samples, 6000 are used for training and 1481 are reserved for evaluation at each checkpoint.

This study conducted seven experiments by varying epochs and batch sizes to evaluate detection accuracy using the Mean Average Precision (mAP) metric. Each experiment produced results in the form of evaluation tables and visualizations, including front-view monocular images and top-view (Bird's Eye View) LiDAR point cloud maps. The goal was to identify the best-performing configuration for detecting three object classes: Car, Cyclist, and Pedestrian.

1. The first result, shown in Table 4 and Figures 6 and 7, involves training the model for 75 epochs with a batch size of 4, totalling 112,500 iterations (calculated as 6000/4 × 75). While the mAP evaluation used a batch size of 8 to store checkpoint results, the training process was interrupted at the 13th epoch due to a "CUDA out of memory" error.

Table 4. Map Evaluation Value Based on Epoch 75, Default Batch Size 4, Batch Size Map 8, Stop Epoch 13

Epoch 75, Batch Size Default 4, Batch Size mAP 8	**Average Precision**			**Mean Average Precision (mAP)**
	Car	***Pedestrian***	***Cyclist***	
epoch-0	0.441	0.117	0.0	0.186
epoch-2	0.788	0.183	0.051	0.341
epoch-4	0.856	0.411	0.064	0.444
epoch-8	0.853	0.372	0.183	0.469
epoch-10	0.875	0.314	0.260	0.483
epoch-12	0.892	0.412	0.197	0.501

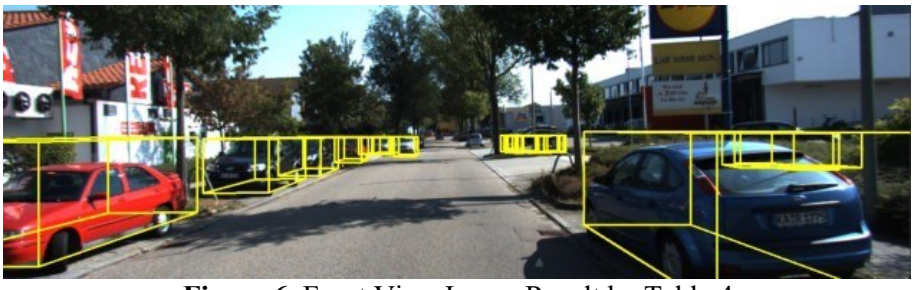

Figure 6. Front View Image Result by Table 4

Figure 7. Bird's Eye View Image Result by Table 4

2. The second result, shown in Table 5 and Figures 8 and 9, involves training the model for 50 epochs with a batch size of 4, totalling 75,000 iterations (calculated as 6000/4 × 50). While the mAP evaluation used a batch size of 8 to store checkpoint results, the training process was interrupted at the 3rd epoch due to a "CUDA out of memory" error.

Table 5. Map Evaluation Value Based on Epoch 50, Default Batch Size 4, Batch Size Map 8, Stop Epoch 3

Epoch 50, Batch Size Default 4, Batch Size mAP 8	**Average Precision**			**Mean Average Precision (mAP)**
	Car	***Pedestrian***	***Cyclist***	
epoch-0	0.621	0.175	0.0	0.265
epoch-2	0.777	0.234	0.067	0.359

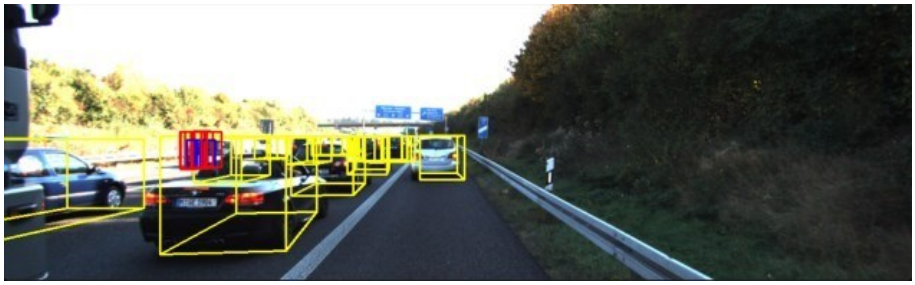

Figure 8. Front View Image Result by Table 5

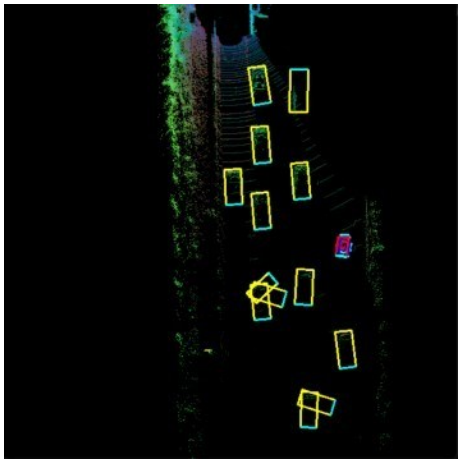

Figure 9. Bird's Eye View Image Result by Table 5

3. The third result, shown in Table 6 and Figures 10 and 11, involves training the model for 50 epochs with a batch size of 2, totalling 150,000 iterations (calculated as 6000/2 × 50). While the mAP evaluation used a batch size of 2 to store checkpoint results, the training process was interrupted at the 3rd epoch due to a "CUDA out of memory" error.

Table 6. Map Evaluation Value Based on Epoch 50, Default Batch Size 2, Batch Size Map 2, Stop Epoch 3

Epoch 50, Batch Size Default 2, Batch Size mAP 2	**Average Precision**			**Mean Average Precision (mAP)**
	Car	***Pedestrian***	***Cyclist***	
epoch-0	0.598	0.152	0.0	0.250
epoch-2	0.781	0.217	0.158	0.385
epoch-4	0.839	0.475	0.120	0.478

Figure 10. Front View Image Result by Table 6

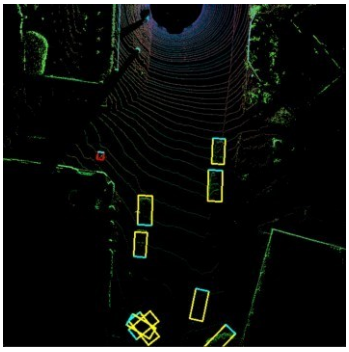

Figure 11. Bird's Eye View Image Result by Table 6

4. The fourth result, shown in Table 7 and Figures 12 and 13, involves training the model for 10 epochs with a batch size of 2, totalling 30,000 iterations (calculated as 6000/2 × 10). While the mAP evaluation used a batch size of 1 to store checkpoint results, and there are no constraint during model training.

Table 7. Map Evaluation Value Based on Epoch 10, Default Batch Size 2, Batch Size Map 1, No Stop Epoch

Epoch 10, Batch Size Default 2, Batch Size mAP 1	**Average Precision**			**Mean Average Precision (mAP)**
	Car	***Pedestrian***	***Cyclist***	
epoch-0	0.565	0.110	0.0	0.225
epoch-2	0.750	0.169	0.005	0.308
epoch-4	0.677	0.302	0.169	0.383
epoch-6	0.817	0.421	0.161	0.466
epoch-8	0.878	0.382	0.338	0.533

Figure 12. Front View Image Result by Table 7

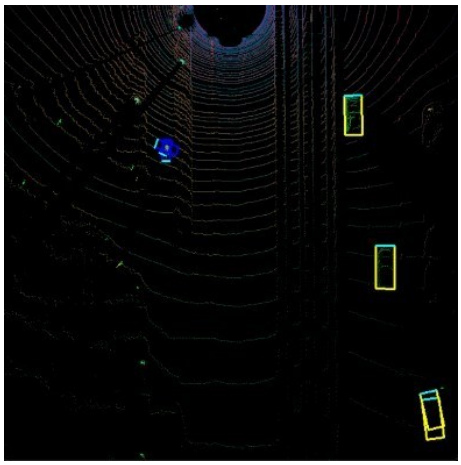

Figure 13. Bird's Eye View Image Result by Table 7

5. The fifth result, shown in Table 8 and Figures 14 and 15, involves training the model for 20 epochs with a batch size of 2, totalling 60,000 iterations (calculated as 6000/2 × 20). While the mAP evaluation used a batch size of 2 to store checkpoint results, the training process was interrupted at the 3rd epoch due to a "CUDA out of memory" error.

Table 8. Map Evaluation Value Based on Epoch 20, Default Batch Size 2, Batch Size Map 2, Stop Epoch 3

Epoch 20, Batch Size Default 2, Batch Size mAP 2	**Average Precision**			**Mean Average Precision (mAP)**
	Car	***Pedestrian***	***Cyclist***	
epoch-0	0.550	0.158	0.0	0.236
epoch-2	0.723	0.196	0.046	0.322

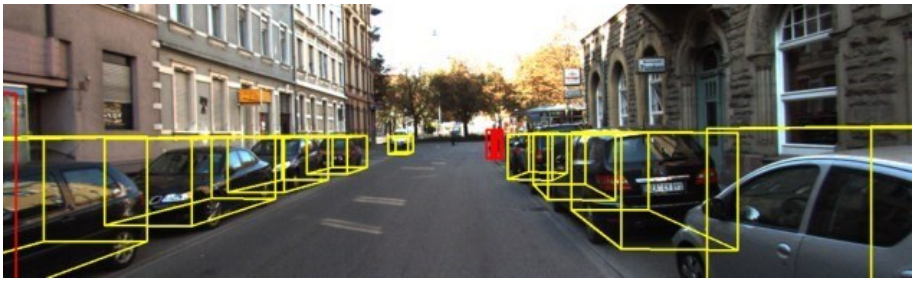

Figure 14. Front View Image Result by Table 8

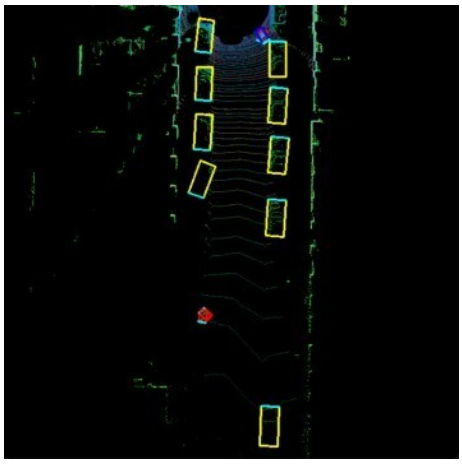

Figure 15. Bird's Eye View Image Result by Table 8

6. The sixth result, shown in Table 9 and Figures 16 and 17, involves training the model for 20 epochs with a batch size of 2, totalling 60,000 iterations (calculated as 6000/2 × 20). While the mAP evaluation used a batch size of 1 to store checkpoint results, and there are no constraint during model training.

Table 9. Map Evaluation Value Based on Epoch 20, Default Batch Size 2, Batch Size Map 1, No Stop Epoch

Epoch 20, Batch Size Default 2, Batch Size mAP 1	**Average Precision**			**Mean Average Precision (mAP)**
	Car	***Pedestrian***	***Cyclist***	
epoch-0	0.539	0.139	0.0	0.226
epoch-2	0.659	0.257	0.019	0.312
epoch-4	0.771	0.292	0.224	0.429
epoch-8	0.817	0.356	0.138	0.437
epoch-10	0.851	0.325	0.155	0.444
epoch-12	0.880	0.424	0.206	0.504
epoch-14	0.907	0.496	0.282	0.562
epoch-16	0.896	0.559	0.333	0.596

Figure 16. Front View Image Result by Table 9

Figure 17. Bird's Eye View Image Result by Table 9

7. The seventh result, shown in Table 10 and Figures 18 and 19, involves training the model for 10 epochs with a batch size of 2, totalling 30,000 iterations (calculated as 6000/2 × 10). While the mAP evaluation used a batch size of 2 to store checkpoint results, the training process was interrupted at the 4th epoch due to a "CUDA out of memory" error.

Table 10. Map Evaluation Value Based on Epoch 10, Default Batch Size 2, Batch Size Map 2, Stop Epoch 4

Epoch 10, Batch Size Default 2, Batch Size mAP 2	**Average Precision**			**Mean Average Precision (mAP)**
	Car	***Pedestrian***	***Cyclist***	
epoch-0	0.294	0.006	0.0	0.100
epoch-2	0.761	0.166	0.029	0.318

Figure 18. Front View Image Result by Table 10

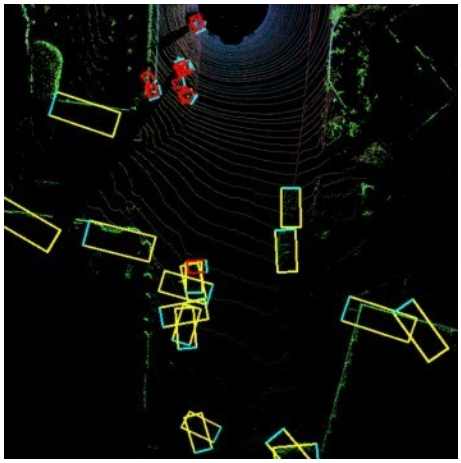

Figure 19. Bird's Eye View Image Result by Table 10

The evaluation indicates that the highest mAP value achieved among the seven training models for 3D object detection using monocular images and LiDAR point clouds is 0.596, rounded to 0.60. This peak performance occurred at the 16th epoch, as detailed in Table 9 and visually represented in chart form in Figure 20.

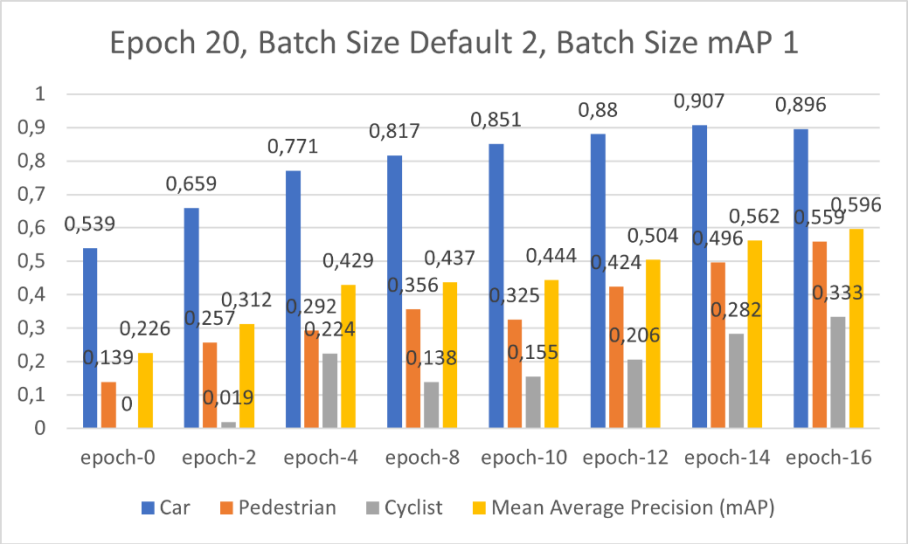

Figure 20. Training Result Chart from Table 9

5. Conclusion

This study successfully applied the YOLOv3 algorithm with a Darknet-53 backbone to perform 3D object detection using monocular camera images and LiDAR point clouds from the KITTI dataset. Despite limitations, it achieved a respectable mAP score of 0.60 at an IoU threshold of 0.5, demonstrating YOLOv3's potential in traffic-based 3D detection tasks.

5.1 Key Contributions and Impact:

The research highlights how sensor fusion—combining monocular and LiDAR data—enhances depth estimation and object detection accuracy. It also shows the practical viability of using affordable sensors for 3D detection, although there is room to improve both hardware and algorithms.

5.2 Limitations:

The study was limited by the KITTI dataset's narrow object range (Car, Cyclist, Pedestrian), sensor performance in adverse conditions (e.g., rain, fog, dust), and the use of the YOLOv3 algorithm, which has since been surpassed by newer models like YOLOv8.

5.3 Recommendations:

Future work should explore larger and more diverse datasets (e.g., Waymo, nuScenes), integrate additional sensors (e.g., radar, thermal cameras), and adopt more advanced detection models such as YOLOv8 or transformer-based methods for improved accuracy and efficiency.

5.4 Accuracy and Improvements:

While the mAP score of 0.60 validates the method's effectiveness, leveraging more advanced algorithms like YOLOv8 could further enhance detection precision and reduce errors.

5.5 Conclusion:

YOLOv3 with sensor fusion provides a solid baseline for 3D object detection in autonomous vehicles. However, future improvements in algorithm selection, dataset diversity, and sensor integration are essential to advance real-world performance and scalability.

References

[1] S. Chen, Y. Li, Z. Lu, R. Li, and G. Chang, "Investigating Multi-Way Impacts of Transportation on Human Footprint: Evidence from China," *Environ Impact Assess Rev*, vol. 98, p. 106896, Jan. 2023, doi: 10.1016/j.eiar.2022.106896.

[2] D. Bonnici and M. Stevens, "It's 2024, how many cars are there in the world?," whichcar. Accessed: Dec. 27, 2024. [Online]. Available: https://www.whichcar.com.au/news/how-many-cars-are-there-in-the-world

[3] World Health Organization, "World Health Organization 2024 data.who.int, Road traffic mortality rate (per 100 000 population)." Accessed: Jan. 06, 2025. [Online]. Available: https://data.who.int/indicators/i/B9D9E6A/D6176E2

[4] Biro Komunikasi dan Informasi Publik, "Tekan Angka Kecelakaan Lalu Lintas, Kemenhub Ajak Masyarakat Beralih ke Transportasi Umum dan Utamakan Keselamatan Berkendara," Kementerian Perhubungan Republik Indonesia. Accessed: Dec. 27, 2024. [Online]. Available: https://dephub.go.id/post/read/%E2%80%8Btekan-angka-kecelakaan-lalu-lintas%2C-kemenhub-ajak-masyarakat-beralih-ke-transportasi-umum-dan-utamakan-keselamatan-berkendara?utm_source=chatgpt.com

[5] K. Bucsuházy, E. Matuchová, R. Zůvala, P. Moravcová, M. Kostíková, and R. Mikulec, "Human factors contributing to the road traffic accident occurrence," *Transportation Research Procedia*, vol. 45, pp. 555–561, Jan. 2020, doi: 10.1016/J.TRPRO.2020.03.057.

[6] P. Liu, R. Yang, and Z. Xu, "How Safe Is Safe Enough for Self-Driving Vehicles?," *Risk Analysis*, vol. 39, no. 2, 2019, doi: 10.1111/risa.13116.

[7] S. Chen, S. Zong, T. Chen, Z. Huang, Y. Chen, and S. Labi, "A Taxonomy for Autonomous Vehicles Considering Ambient Road Infrastructure," *Sustainability*, vol. 15, no. 14, p. 11258, Jul. 2023, doi: 10.3390/su151411258.

[8] E. Arnold, O. Y. Al-Jarrah, M. Dianati, S. Fallah, D. Oxtoby, and A. Mouzakitis, "A Survey on 3D Object Detection Methods for Autonomous Driving Applications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 10, pp. 3782–3795, Oct. 2019, doi: 10.1109/TITS.2019.2892405.

[9] K. F. Yuen, Y. D. Wong, F. Ma, and X. Wang, "The Determinants of Public Acceptance of Autonomous Vehicles: An Innovation Diffusion Perspective," *J Clean Prod*, vol. 270, p. 121904, Oct. 2020, doi: https://doi.org/10.1016/j.jclepro.2020.121904.

[10] Siti Nur Arifa, "Mengenal Level Kendaraan Otonom dan Peluang Beroperasi Massal di Indonesia," Good News. Accessed: Dec. 18, 2024. [Online]. Available: https://www.goodnewsfromindonesia.id/2022/05/21/mengenal-level-kendaraan-otonom-dan-peluang-beroperasi-massal-di-indonesia

[11] M. Simon, S. Milz, K. Amende, and H.-M. Gross, "Complex-YOLO: Real-time 3D Object Detection on Point Clouds," Mar. 2018, [Online]. Available: http://arxiv.org/abs/1803.06199

[12] B. L. Wibowo, O. Y. Caesarizky, M. I. Hakim, and M. Munsarif, "Penerapan Algoritma Convolutional Neural Network (CNN) untuk Mengklasifikasi Jenis Kendaraan," *JURNAL KOMPUTER DAN TEKNOLOGI INFORMASI*, vol. 2, no. 2, pp. 94–99, Jul. 2024, doi: 10.26714/jkti.v%vi%i.13938.

[13] M. Simon, S. Milz, K. Amende, and H. M. Gross, "Complex-YOLO: An Euler-Region-Proposal for Real-Time 3D Object Detection on Point Clouds," in *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, ECCV 2018 Workshops, Sep. 2018, pp. 0–0. doi: 10.1007/978-3-030-11009-3_11.

[14] X. Chen, K. Kundu, Z. Zhang, H. Ma, S. Fidler, and R. Urtasun, "Monocular 3D Object Detection for Autonomous Driving," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA: IEEE, Dec. 2016, pp. 2147–2156. doi: 10.1109/CVPR.2016.236.

[15] H. Liu, Y. Wang, H. Chu, B. Zou, J. Chen, and H. Ma, "Enhancing pseudo label quality for pedestrian and cyclist in weakly supervised 3D object detection," *Neurocomputing*, vol. 584, p. 127531, Jun. 2024, doi: 10.1016/j.neucom.2024.127531.

[16] T. Karim, Z. R. Mahayuddin, and M. K. Hasan, "Singular and Multimodal Techniques of 3D Object Detection: Constraints, Advancements and Research Direction," *Applied Sciences*, vol. 13, no. 24, p. 13267, Dec. 2023, doi: 10.3390/app132413267.

[17] R.-A. Bratulescu, R.-I. Vatasoiu, G. Sucic, S.-A. Mitroi, M.-C. Vochin, and M.-A. Sachian, "Object Detection in Autonomous Vehicles," in *2022 25th International Symposium on Wireless Personal Multimedia Communications (WPMC)*, Herning, Denmark: IEEE, Oct. 2022, pp. 375–380. doi: 10.1109/WPMC55625.2022.10014804.

[18] M. Y. Ridho, "Dari YOLO ke YOLOv8: Menelusuri Evolusi Algoritma Deteksi Objek Perbaikan apa yang dilakukan dalam tujuh tahun terakhir?," Medium. Accessed: Jan. 21, 2025. [Online]. Available: https://medium.com/%40muhammadyusufridho6/dari-yolo-ke-yolov8-menelusuri-evolusi-algoritma-deteksi-objek-9b714fd66b78

[19] M. A. R. Alif and M. Hussain, "YOLOv1 to YOLOv10: A Comprehensive Review of YOLO Variants and Their Application in the Agricultural Domain," *Department of Computer Science, Huddersfield University, Queensgate, Huddersfield HD1 3DH, UK*, pp. 1–31, Jun. 2024, doi: https://doi.org/10.48550/arXiv.2406.10139.

[20] J. P. Ortiz, J. D. Valladolid, and D. Dutan, "A Comparative Analysis for Traffic Officer Detection in Autonomous Vehicles using YOLOv3, v5, and v8," in *2024 IEEE Eighth Ecuador Technical Chapters Meeting (ETCM)*, Cuenca, Ecuador: IEEE, Oct. 2024, pp. 1–7. doi: 10.1109/ETCM63562.2024.10746133.

[21] A. Geiger, P. Lenz, and R. Urtasun, "Are We Ready for Autonomous Driving? The KITTI Vision Benchmark Suite," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, Providence, RI, USA: IEEE, Jul. 2012. doi: 10.1109/CVPR.2012.6248074.

[22] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," Jan. 2018, doi: 10.48550/arXiv.1804.02767.

[23] D. Nafis Alfarizi, R. Agung Pangestu, D. Aditya, M. Adi Setiawan, and P. Rosyani, "Penggunaan Metode YOLO Pada Deteksi Objek: Sebuah Tinjauan Literatur Sistematis," *Jurnal Artificial Inteligent dan Sistem Penunjang Keputusan*, vol. 1, no. 1, pp. 54–63, Jun. 2023, Accessed: Jan. 08, 2025. [Online]. Available: http://jurnalmahasiswa.com/index.php/aidanspk/article/view/144

[24] A. Chazhoor and V. Sarobin, "Intelligent Automation of Invoice Parsing Using Computer Vision Techniques," *Multimed Tools Appl*, vol. 81, no. 1, pp. 29383–29403, Aug. 2022, doi: 10.1007/s11042-022-12916-x.

[25] H. Li *et al.*, "3DSG: A 3D LiDAR-Based Object Detection Method for Autonomous Mining Trucks Fusing Semantic and Geometric Features," *Applied Sciences*, vol. 12, no. 23, p. 12444, Dec. 2022, doi: 10.3390/app122312444.

[26] H. Wang, Y. Cong, O. Litany, Y. Gao, and L. J. Guibas, "3Dioumatch: Leveraging IoU prediction for semi-supervised 3D object detection," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul. 2021, pp. 14615–14624. doi: 10.1109/CVPR46437.2021.01438.