engineering proceedings

MDPI

Proceeding Paper

Using a Monocular Camera for 360° Dynamic Object Instance Segmentation in Traffic †

Goran Oreski * and Lucija Babic

Faculty of Informatics, Juraj Dobrila University of Pula, Zagrebacka 30, 52100 Pula, Croatia; lucija.babic@student.unipu.hr

* Correspondence: goreski@unipu.hr

† Presented at the International Conference on Electronics, Engineering Physics and Earth Science (EEPES 2023), Kavala, Greece, 21–23 June 2023.

Abstract: As autonomous driving is quickly becoming a reality, the most considerable challenge with self-driving remains safety, as the car must process surroundings to make the correct predictions. This paper explores the detection and classification of moving objects, such as vehicles and pedestrians, on a pixel level. Using the CARLA simulator, we collected data from a single front-facing camera, thus excluding standard sensor technologies like lidar or radar. This research aims to test the generalization capabilities of models trained using front-camera images on inputs from differently positioned cameras capturing 360 surroundings. We tackle the described challenge with the task of instance segmentation using the most famous instance segmentation algorithms: Mask Region-based Convolutional Neural Network and YOLOv7. We show to what extent a monocular setup can be used for the instance segmentation of the perspectives that differ from the baseline captured by the front camera.

Keywords: dynamic traffic objects; vehicles; pedestrians; instance segmentation; autonomous driving; YOLO; Mask R-CNN

check for updates

Citation: Oreski, G.; Babic, L. Using a Monocular Camera for 360° Dynamic Object Instance Segmentation in Traffic. *Eng. Proc.* **2023**, *41*, 6. https://doi.org/10.3390/engproc2023041006

Academic Editor: Grigor Mihaylov

Published: 13 July 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Nowadays, it is a common occurrence for vehicles to be equipped with built-in cameras. The cameras serve a variety of purposes, from assisting drivers during parking and reversing to providing security footage. The integration of camera systems has promoted the development of advanced driver assistance systems, offering drivers warnings and supportive features for a safer driving experience [1]. The implementation of such systems significantly improves traffic safety, as human error continues to be the leading cause of driving accidents, as reported by the European Commission (2019, 2020).

All of these technological advancements have paved the way for the development of the most complex systems that have enormous potential for transforming transportation in the future—autonomous driving systems. Therefore, in the near future, it is expected that there will be widespread deployment of autonomous vehicles (AVs) on urban roads [2]. One of the most significant challenges in developing any production-ready deep learning model is the quality and amount of data. That especially holds for AVs, where a tremendous amount of funds are allocated for collecting rich real-life datasets. The companies record traffic using different sensors collecting petabytes of data. Collecting and storing that amount, as well as the sensors used for recording, is expensive.

This study investigates to what extent a monocular, front-facing camera can be used for training models that, in inference time, take an input of the full view of the vehicle's surroundings, as shown in Figure 1. We test the generalization capabilities of models trained solely on front-camera images on additional, differently facing cameras that cover a 360 view. The research analyzes how the change in perspective from different camera

orientations affects performance. The findings answer how the recordings from front cameras, widely available on the internet, can be reused in training AV models that have to capture the entire car's environment, thus reducing the cost associated with data collection. We focus on a dynamic traffic environment, vehicles and pedestrians, and compare the performance of the Mask R-CNN [3] and YOLO [4] algorithms.

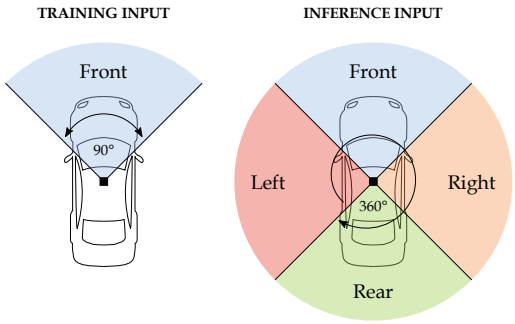


Figure 1. Graphic display of camera positions.

This paper is organized as follows: Section 2 provides a comprehensive review of the existing literature on instance segmentation and cameras on autonomous vehicles accompanied by a discussion of their relevance to our problem. In Section 3, we present an in-depth analysis of two different instance segmentation models, highlighting their differences and comparing their performance on a common dataset. Section 4 presents our research methodology, including details on the CARLA Simulator, data extraction, and data preprocessing and training. In Section 5, we evaluate the performance of the trained models on 360-degree field-of-view data, with a specific focus on comparing their performance on front-facing camera images versus data from the left, right, and rear cameras. Finally, in Section 6, we present our conclusions and suggest areas for future research to improve upon our findings.

2. Literature Review

With the rapid development of deep neural network architectures [5] comes the task of instance segmentation [6]. This task falls under the field of computer vision [7]. It involves identifying individual objects within an image and labeling each pixel in the image with a unique object ID. In other words, instance segmentation is a combination of object detection and semantic segmentation. Over the years, with advancements in computer vision technology, the performance of instance segmentation has greatly improved. Instance segmentation models are already used to identify and segment various objects [8–10], resulting in the automation of processes in various fields.

In the context of autonomous driving, instance segmentation is an active area of research. One of the main fields of application is lane and road marking detection, where instance segmentation is used to improve detecting precision [11–13]. Similarly to our research, other papers have applied instance segmentation on dynamic traffic elements like vehicles [14,15] and pedestrians [16,17]. For example, the authors in [18] propose an implementation of a state-of-the-art Mask R-CNN method using a transfer learning technique for vehicle detection via instance-wise segmentation, which produces a bounding box and object mask simultaneously. The authors in [16] also applied the Mask R-CNN based framework, for instance segmentation applied to pedestrian crosswalks, and achieved impressive accuracy. In [19], a fast one-stage multi-task neural network is proposed, for instance segmentation, which can meet the requirements of real-time processing with sufficient accuracy, an important feature for self-driving applications. The third direction of the research is related to the sensors used on a vehicle; some papers use only cameras [20,21]; others include lidar [22,23]. And significant effort is directed toward fusing data collected by different sensor types [24,25].

Our research uses camera sensors and dynamic objects, i.e., vehicles and pedestrians. Our main goal is to discover the generalization capabilities of the most popular algorithms on images that capture a slightly different perspective from the one on which the model is trained.

3. Architectures

Mask R-CNN and YOLO differ fundamentally as they represent the two predominant classes of object detectors in the field of computer vision: the former being a two-stage detector that generates region proposals and classifies objects within them, whereas the latter takes a single-stage approach by treating the problem as a regression task to predict bounding box coordinates and class probabilities directly from the image [26].

3.1. Mask R-CNN

Mask R-CNN is a state-of-the-art deep learning algorithm for object detection and instance segmentation. It extends the popular Faster R-CNN framework by adding an additional branch to the network that outputs a binary mask for each detected object, indicating which pixels belong to the object and which do not.

Mask R-CNN uses a two-stage architecture, similar to Faster R-CNN. In the first stage, the network generates a set of region proposals using a Region Proposal Network (RPN). These proposals are then fed into the second stage, where the network performs classification, bounding box regression, and mask prediction.

The mask branch of Mask R-CNN is a fully convolutional network that takes the region of interest (ROI) as input and generates a binary mask for each object in the ROI. This mask is used to segment the object from the background and obtain more precise boundaries around the object.

One of the key benefits of Mask R-CNN is that it can perform object detection and instance segmentation simultaneously, which is useful in applications such as autonomous driving, medical imaging, and robotics. The algorithm has achieved state-of-the-art performance on several benchmark datasets, including COCO and Cityscapes.

3.2. YOLO Series

You Only Look Once (YOLO) is a single-stage deep learning algorithm for object detection in images and videos. It is based on convolutional neural networks (CNNs) and uses a single neural network to predict both the locations and classes of objects in an image.

Unlike traditional object detection algorithms that use a sliding-window approach to classify regions of an image, YOLO divides an image into a grid of cells and makes predictions based on the contents of each cell. For each cell in the grid, YOLO predicts the class probabilities for each object that is present in the cell, as well as the bounding boxes that enclose the objects.

To train the YOLO algorithm, a large dataset of labeled images is used. The network is trained to predict the correct bounding boxes and class probabilities for each object in the image. During training, the network's parameters are adjusted using backpropagation to minimize the prediction error.

One of the benefits of YOLO is its real-time performance. Because it processes the entire image in a single pass, it can be used for applications such as autonomous vehicles, drones, and video surveillance in real-time. Additionally, YOLO can detect objects of different sizes and shapes, making it suitable for a wide range of applications.

The YOLO algorithm has undergone rapid development, with each new version building upon its predecessors through ongoing improvements [27]. The recent state of-the-art YOLO version 7 [28] outperforms all existing object detection models in terms of speed and accuracy, while using fewer parameters and less computation. YOLOv7 has a faster and stronger network architecture that integrates features effectively, provides more accurate object detection and instance segmentation performance, and has a more robust

loss function and improved training efficiency. This makes YOLOv7 more cost-effective and able to train faster on smaller datasets without pretrained weights.

4. Research Methodology

This section provides a comprehensive overview of our experiment, shown in Figure 2.

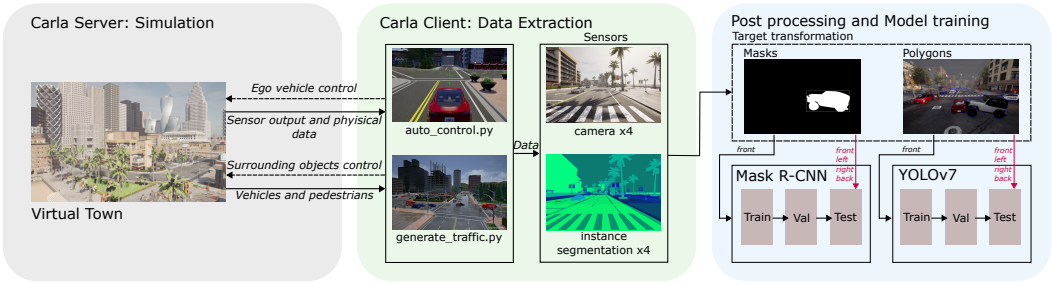


Figure 2. Graphic representation of the experiment.

4.1. CARLA Simulator and Data Extraction

To develop the models, we first need sufficient data to train, validate, and test. We obtained the data using the CARLA simulator [29], an open-source simulator designed for autonomous driving research, specifically to facilitate the development, training, and validation of autonomous driving systems. The platform offers open digital assets like urban layouts, buildings, and vehicles and features such as specifying sensor suites and environmental conditions, full control over static and dynamic actors, and the ability to generate maps. We used the simulator to capture high-quality RGB images with front-, left-, right-, and rear-facing cameras, each providing a 90-degree field of view.

We generated around a hundred vehicles and pedestrians each to roam the streets of the default town freely. Next, we initialized the sensors on a moving car to collect the data. While the CARLA blueprint library offers a wide range of sensors, we needed RGB and instance segmentation cameras. To set up cameras, we spawned and attached four different point-of-view cameras for two types of sensors on our Ego vehicle, which we set to drive around town in autopilot mode. Once everything was set up, all sensors waited for a signal of frame change, with a lag of 10 frames, to capture images and save them to the disk, creating the annotated dataset. Distinguishing between different instances of pedestrians and vehicles in the final dataset was straightforward. The instance segmentation image, saved to disk, had the instance IDs encoded in the G and B channels of the RGB image file, while the R channel contained the standard class ID. Pedestrians and vehicles have class IDs of 4 and 10, respectively, making preparing targets for our models reasonably simple.

4.2. Data Preprocessing and Training

To create a Mask R-CNN target for each training instance, we needed to generate binary masks for each object in an image. Fortunately, we could leverage the instance IDs in the G and B channels of the instance segmentation image to separate objects. Each mask represents a single object in one channel image with the same resolution as the original input. The binary mask contains only ones and zeroes, indicating the pixels belonging to a particular object. Simultaneously, we produced a list of class IDs, where 1 corresponds to pedestrians and 2 to vehicles; class 0 was used as the background. Additionally, we generated the bounding box coordinates and calculated the corresponding surface area for each instance. The surface area was used to filter the noise, i.e., low-pixel objects. After the described process, the Mask R-CNN dataset was ready.

The YOLO algorithm targets significantly differ. Rather than requiring a mask for each instance, YOLO takes a bounding polygon. A polygon is a set of an arbitrary number of points creating a complex shape around an object. Examples of masks and polygons are depicted in Figure 2.

Table 1 lists the basic configuration and hyperparameters of models used in the experiment. The parameters were kept default as defined in the original repositories. The experiment's goal was not to achieve state-of-the-art performance but to compare the generalization capabilities of two architectures from unseen perspectives. The generalization is defined as the performance difference between instance segmentation on the front camera (baseline) and instance segmentation on differently facing cameras.

Table 1. Values of key hyperparameters used in training.

Name	Mask R-CNN [30]	YOLOv7 [28]
Optimizer	Adam	SGD+M
Learning rate	1×10^{-4}	1×10^{-3}
Number of epochs	32	30
Pretrained weights	None	COCO
Backbone	Resnet50	E-ELAN
Batch size	2	4

The generated dataset of 10,000 images was used to train both models, which was subsequently split into train and validation sets according to an 80:20 ratio. Notably, the models were trained using images captured solely by a front-facing camera, whereas the test process leveraged a full 360-degree view of a vehicle's surroundings. Following the training of each model, a set of 1000 unseen test images was used to evaluate performance. Upon initial examination, as shown in Figure 3, both models produced comparable and accurate outcomes, correctly segmenting objects captured by side and rear cameras, including vehicles and pedestrians. Models used output segmentation masks and bounding boxes for each input image.



Figure 3. Display of the results on the same 360° scene.

5. Results and Discussion

We will use a standard metric for evaluating the computer vision deep learning models, *Mean Average Precision* (mAP), to evaluate the results. It measures how well the system matches the predicted bounding boxes to the ground-truth bounding boxes for each class of object, using the *Intersection over Union* (IoU) metric to calculate the overlap between the boxes. The mAP is calculated by computing the average of the area under the precision–recall curve for each class. A higher mAP score indicates better accuracy of the object detection system. This section will analyze the mAP results using an exact 0.5 threshold

and varying IoU threshold, averaging multiple thresholds ranging from 0.5 to 0.95. We will apply both measures to the bounding box results marked as *Boxset* in the results, which represent the performance of object detection, and to the mask predictions marked as *Maskset*, which pertain to the accuracy of instance segmentation. Since our objective is to evaluate the latter, we will mainly focus on analyzing the *Maskset* results.

5.1. Results

Table 2 presents the results of the models on different input perspectives. First, we analyze how the models perform on front camera images, the baseline perspective they were trained on. YOLOv7 is more accurate than Mask R-CNN since it has higher overall mAP results. However, there is a noticeable difference in the consistency between box and mask predictions on the mAP .5 between the architectures. While Mask R-CNN detects and segments objects at approximately 50% mAP, YOLOv7 performs better object detection than instance segmentation. Such results confirm that YOLO is primarily a detection algorithm. Although YOLOv7 demonstrates superior performance, it drops in instance segmentation, whereas Mask R-CNN remains consistent in detection and segmentation.

Next, we focus on the models' performance on cameras with perspectives that differ from the baseline, specifically mask prediction (*Maskset*), to evaluate how well the models perform instance segmentation. We observe interesting results. Mask R-CNN has better mAP .5 on the left and right cameras, despite YOLOv7 having better baseline results, meaning Mask R-CNN has a considerably better generalization on unseen side perspectives than YOLOv7. If we analyze mAP .5:.95 on the same side cameras, even though the absolute values are similar, YOLO significantly drops in performance from baseline. The results for the rear camera, which has a different perspective than the front camera but can be considered a reflection of the training inputs, favor YOLOv7. Given this characteristic, both models can generalize well on the rear camera images.

Table 2. The results of the experiment for all classes.

Model	Camera	Boxset		Maskset	
		mAP .5	mAP .5:.95	mAP .5	mAP .5:.95
Mask R-CNN	Front (baseline)	0.526	0.309	0.502	0.239
	Left	0.439	0.213	0.396	0.162
	difference	−0.087	−0.096	−0.106	−0.077
	Right	0.462	0.247	0.459	0.202
	difference	−0.064	−0.062	−0.043	−0.037
	Rear	0.406	0.209	0.389	0.18
	difference	−0.12	−0.1	−0.113	−0.059
Yolov7	Front (baseline)	0.687	0.494	0.59	0.305
	Left	0.402	0.251	0.331	0.172
	difference	−0.285	−0.243	−0.259	−0.133
	Right	0.45	0.289	0.382	0.183
	difference	−0.237	−0.205	−0.208	−0.122
	Rear	0.597	0.371	0.483	0.257
	difference	−0.09	−0.123	−0.107	−0.048

5.2. Results for Vehicle Segmentation

To better understand the results, we provide the performance by each class individually. First, we focus on vehicles; the results are listed in Table 3. Vehicles generated by CARLA range from different kinds of cars and trucks to smaller vehicles like motorcycles and bicycles. Vehicles can be positioned closer and further away on the horizon. If a vehicle is close, usually it means the *Ego* car is driving behind it in the line and collecting images of the rear end of vehicles. In most cases, a front camera can record the front and left side of a vehicle when it is further away, especially if the car is driving in the opposite direction, but in rare cases, we see the right vehicle side. These points are essential to make

so we can understand the data our models are trained on and how they may differ from other perspectives.

YOLOv7 performs better in both tasks and exceptionally well in vehicle detection. Both models report a performance drop regarding the *Maskset* generalization capabilities on left and right cameras. Such behavior can be attributed to the lack of a close-up view of the sides of vehicles since the front camera does not commonly capture them. The dataset considerations, discussed earlier, explain why the right perspective performs the worst. Although Mask R-CNN performs closer to the baseline than YOLOv7, it is less accurate. The results for the rear camera are expected since the rear camera mainly captures the front of a vehicle up close, while the distant view is mirrored to that of the front camera. YOLOv7 is more accurate at segmenting vehicles from the rear perspective than Mask R-CNN and is closer to the baseline, as seen in both mAP metrics.

Table 3. The results of the experiment for vehicles.

Model	Camera	Boxset		Maskset	
		mAP .5	mAP .5:.95	mAP .5	mAP .5:.95
Mask R-CNN	Front (baseline)	0.67	0.448	0.636	0.361
	Left difference	0.452 −0.218	0.263 −0.185	0.437 −0.199	0.236 −0.125
	Right difference	0.381 −0.289	0.215 −0.233	0.386 −0.25	0.204 −0.157
	Rear difference	0.48 −0.19	0.284 −0.164	0.485 −0.151	0.275 −0.086
Yolov7	Front (baseline)	0.83	0.641	0.735	0.438
	Left difference	0.52 −0.31	0.346 −0.295	0.469 −0.266	0.275 −0.163
	Right difference	0.411 −0.419	0.28 −0.361	0.377 −0.358	0.23 −0.208
	Rear difference	0.733 −0.097	0.503 −0.138	0.659 −0.076	0.395 −0.043

5.3. Results for Pedestrian Segmentation

Table 4 contains the results for pedestrians. Pedestrians are mainly visible on the sidewalk in the images captured from the front camera but could also be present at the crossroad. Notably, pedestrians can usually be spotted at a considerable distance from our viewpoint, implying that our models are trained to recognize relatively small pedestrian features. Regarding the overall baseline performance, YOLOv7 again outperforms Mask R-CNN.

The left camera usually captures people on the left sidewalk. Based on this observation, the left camera typically records images of pedestrians in higher resolution. Mask R-CNN does not lose mAP performance as it did in vehicle segmentation. It outperforms YOLOv7 in detection and segmentation. The difference in the prediction performance of Mask R-CNN for pedestrians from baseline in terms of detection is significantly positive and almost the same in terms of instance segmentation. Therefore, the results indicate that Mask R-CNN is exceptionally good at learning pedestrian features from that perspective. In contrast, YOLOv7's performance decreased significantly compared to the baseline.

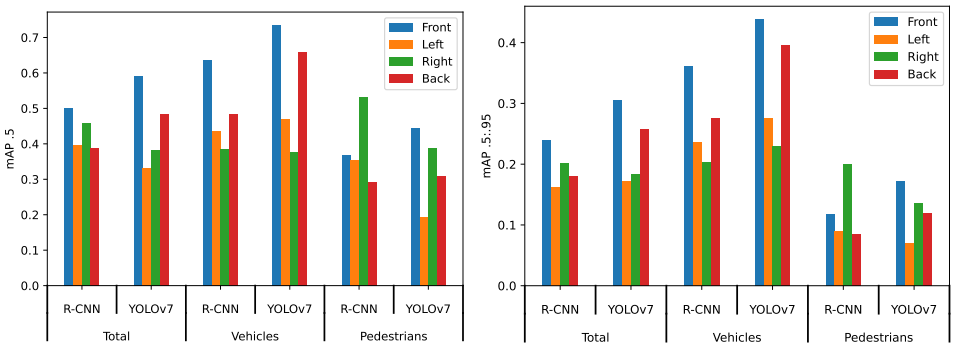
The right camera observes pedestrians walking on the right sidewalk. This implies that the right camera is the one that captures pedestrians at a much larger size, allowing for a better recognition of features. It can be noticed that Mask R-CNN improves the detection and segmentation mAP baseline results by a significant margin. All differences from the baseline are positive, and if we focus on segmentation, we observe an increase of 0.16 percentage points in mAP .5. YOLOv7's performance again drops across all evaluation measures.

As for the rear camera, the following results were observed: Since the viewpoint is the same as the front camera but mirrored, we anticipated that the pedestrian segmentation results would be similar. YOLOv7 is once again more accurate than Mask R-CNN, but the difference is less significant than with vehicle segmentation. Comparing the difference in pedestrian segmentation from the baseline, Mask R-CNN has better generalization (smaller difference) than YOLOv7 for the first time.

Table 4. The results of the experiment for pedestrians.

Model	Camera	Boxset		Maskset	
		mAP .5	mAP .5:.95	mAP .5	mAP .5:.95
Mask R-CNN	Front (baseline)	0.383	0.17	0.368	0.117
	Left	0.426	0.163	0.355	0.089
	difference	0.043	−0.007	−0.013	−0.028
	Right	0.544	0.278	0.532	0.2
	difference	0.161	0.108	0.164	0.083
	Rear	0.332	0.135	0.293	0.084
	difference	−0.051	−0.035	−0.075	−0.033
Yolov7	Front (baseline)	0.543	0.348	0.444	0.172
	Left	0.283	0.156	0.194	0.0697
	difference	−0.26	−0.192	−0.25	−0.1023
	Right	0.488	0.298	0.387	0.136
	difference	−0.055	−0.05	−0.057	−0.036
	Rear	0.46	0.239	0.308	0.119
	difference	−0.083	−0.109	−0.136	−0.053

In Figure 4, a graphical representation of the instance segmentation results is presented in terms of the mAP .5 and mAP .5:.95 metrics. From the Figure 4, it is clear that YOLOv7 has higher mAP performance in most cases. Regarding vehicle segmentation, both models perform similarly for the left and right perspectives, even though YOLOv7's baseline results are better. Mask R-CNN is the only architecture with positive differences compared to the baseline, as shown in the pedestrian results. If we examine the right camera specifically, Mask R-CNN was able to recognize pedestrians that are very close to the viewpoint exceptionally well. In contrast, YOLOv7 fails to perform better than the baseline when presented with up-close pedestrians. On the other hand, YOLOv7 has better generalization on the rear camera.



(**a**) (**b**)

Figure 4. Instance segmentation results (*Maskset*) measured by mAP using two tresholds (**a**) mAP .5 and (**b**) mAP .5:.95.

6. Conclusions

This paper explored the generalization properties of two popular instance segmentation algorithms on unseen perspectives of moving objects in traffic. The goal was to

establish whether the pervasive front-facing camera captures can be used as training data for a full surrounding view of a vehicle. Specifically, the generalization capabilities, defined as the difference in mAP measure between the baseline (front camera) and other perspectives (left, right, and rear cameras), were compared.

From the results, YOLOv7 performs better on the front camera, i.e., produces more accurate baselines. That is probably because YOLOv7 extensively uses augmentation techniques, while Mask R-CNN used in this experiment does not. The experiment's main aim was not to compare absolute mAP values but the generalization performance as described earlier. YOLOv7 also typically has better generalization results on the rear camera tests. Therefore, if the perspective data do not deviate far from the data used in training, the YOLOv7 architecture is a better choice. However, at the same time, it fails to extract meaningful features from larger, differently oriented objects that appeared smaller in training. On the other hand, Mask R-CNN has better generalization in side perspectives, meaning its performance drops less on images that differ more from the base one. In the case of pedestrian segmentation, it even improves baseline performance.

In future work, it would be interesting to add images from one additional side-facing camera in the training phase and test the generalization capabilities of the YOLO algorithm.

Author Contributions: Conceptualization, G.O. and L.B.; methodology, G.O.; software, G.O. and L.B.; validation, G.O. and L.B.; formal analysis, G.O. and L.B.; investigation, G.O. and L.B.; resources, G.O. and L.B.; data curation, L.B.; writing—original draft preparation, G.O. and L.B.; writing—review and editing, G.O. and L.B.; visualization, G.O. and L.B.; supervision, G.O.; project administration, G.O.; funding acquisition, G.O. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Chiang, Y.M.; Hsu, N.Z.; Lin, K.L. Driver assistance system based on monocular vision. In *Proceedings of the New Frontiers in Applied Artificial Intelligence: 21st International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, IEA/AIE 2008 Wrocław, Poland, 18–20 June 2008*; Springer: Berlin/Heidelberg, Germany, 2008; pp. 1–10.
2. Chen, J.; Zhao, C.; Jiang, S.; Zhang, X.; Li, Z.; Du, Y. Safe, Efficient, and Comfortable Autonomous Driving Based on Cooperative Vehicle Infrastructure System. *Int. J. Environ. Res. Public Health* **2023**, *20*, 893. [CrossRef] [PubMed]
3. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2961–2969.
4. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
5. Liu, W.; Wang, Z.; Liu, X.; Zeng, N.; Liu, Y.; Alsaadi, F.E. A survey of deep neural network architectures and their applications. *Neurocomputing* **2017**, *234*, 11–26. [CrossRef]
6. Hafiz, A.M.; Bhat, G.M. A survey on instance segmentation: State of the art. *Int. J. Multimed. Inf. Retr.* **2020**, *9*, 171–189. [CrossRef]
7. Voulodimos, A.; Doulamis, N.; Doulamis, A.; Protopapadakis, E. Deep learning for computer vision: A brief review. *Comput. Intell. Neurosci.* **2018**, *2018*, 7068349. [CrossRef] [PubMed]
8. Jaikumar, P.; Vandaele, R.; Ojha, V. Transfer learning for instance segmentation of waste bottles using Mask R-CNN algorithm. In *Proceedings of the Intelligent Systems Design and Applications: 20th International Conference on Intelligent Systems Design and Applications (ISDA 2020), Online, 12–15 December 2020*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 140–149.
9. Perez, M.I.; Karelovic, B.; Molina, R.; Saavedra, R.; Cerulo, P.; Cabrera, G. Precision silviculture: Use of UAVs and comparison of deep learning models for the identification and segmentation of tree crowns in pine crops. *Int. J. Digit. Earth* **2022**, *15*, 2223–2238. [CrossRef]
10. Polewski, P.; Shelton, J.; Yao, W.; Heurich, M. Instance segmentation of fallen trees in aerial color infrared imagery using active multi-contour evolution with fully convolutional network-based intensity priors. *ISPRS J. Photogramm. Remote Sens.* **2021**, *178*, 297–313. [CrossRef]

11. Ko, Y.; Lee, Y.; Azam, S.; Munir, F.; Jeon, M.; Pedrycz, W. Key points estimation and point instance segmentation approach for lane detection. *IEEE Trans. Intell. Transp. Syst.* **2021**, *23*, 8949–8958. [CrossRef]
12. Zhang, H.; Luo, G.; Tian, Y.; Wang, K.; He, H.; Wang, F.Y. A virtual-real interaction approach to object instance segmentation in traffic scenes. *IEEE Trans. Intell. Transp. Syst.* **2020**, *22*, 863–875. [CrossRef]
13. Chang, D.; Chirakkal, V.; Goswami, S.; Hasan, M.; Jung, T.; Kang, J.; Kee, S.C.; Lee, D.; Singh, A.P. Multi-lane detection using instance segmentation and attentive voting. In Proceedings of the 2019 19th International Conference on Control, Automation and Systems (ICCAS), Jeju, Republic of Korea, 15–18 October 2019; pp. 1538–1542.
14. Zhang, B.; Zhang, J. A traffic surveillance system for obtaining comprehensive information of the passing vehicles based on instance segmentation. *IEEE Trans. Intell. Transp. Syst.* **2020**, *22*, 7040–7055. [CrossRef]
15. de Carvalho, O.L.F.; de Carvalho Júnior, O.A.; de Albuquerque, A.O.; Santana, N.C.; Guimarães, R.F.; Gomes, R.A.T.; Borges, D.L. Bounding box-free instance segmentation using semi-supervised iterative learning for vehicle detection. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2022**, *15*, 3403–3420. [CrossRef]
16. Malbog, M.A. MASK R-CNN for pedestrian crosswalk detection and instance segmentation. In Proceedings of the 2019 IEEE 6th International Conference on Engineering Technologies and Applied Sciences (ICETAS), Kuala Lumpur, Malaysia, 20–21 December 2019; pp. 1–5.
17. Lyssenko, M.; Gladisch, C.; Heinzemann, C.; Woehrle, M.; Triebel, R. Instance Segmentation in CARLA: Methodology and Analysis for Pedestrian-oriented Synthetic Data Generation in Crowded Scenes. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, QC, Canada, 10–17 October 2021; pp. 988–996.
18. Ojha, A.; Sahu, S.P.; Dewangan, D.K. Vehicle detection through instance segmentation using mask R-CNN for intelligent vehicle system. In Proceedings of the 2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, 6–8 May 2021; pp. 954–959.
19. Tseng, K.K.; Lin, J.; Chen, C.M.; Hassan, M.M. A fast instance segmentation with one-stage multi-task deep neural network for autonomous driving. *Comput. Electr. Eng.* **2021**, *93*, 107194. [CrossRef]
20. Deng, Z.; Chen, Y.; Liu, L.; Wang, S.; Ke, R.; Schonlieb, C.B.; Aviles-Rivero, A.I. TrafficCAM: A Versatile Dataset for Traffic Flow Segmentation. *arXiv* **2022**, arXiv:2211.09620.
21. Zhang, T.; Jin, P.J.; Ge, Y.; Moghe, R.; Jiang, X. Vehicle detection and tracking for 511 traffic cameras with U-shaped dual attention inception neural networks and spatial-temporal map. *Transp. Res. Rec.* **2022**, *2676*, 613–629.
22. Rotter, P.; Klemiato, M.; Skruch, P. Automatic Calibration of a LiDAR–Camera System Based on Instance Segmentation. *Remote Sens.* **2022**, *14*, 2531. [CrossRef]
23. Li, X.; Yin, J.; Shi, B.; Li, Y.; Yang, R.; Shen, J. LWSIS: LiDAR-guided Weakly Supervised Instance Segmentation for Autonomous Driving. *arXiv* **2022**, arXiv:2212.03504.
24. Jiang, Q.; Sun, H.; Zhang, X. SemanticBEVFusion: Rethink LiDAR-Camera Fusion in Unified Bird's-Eye View Representation for 3D Object Detection. *arXiv* **2022**, arXiv:2212.04675.
25. Wang, M.; Zhao, L.; Yue, Y. PA3DNet: 3-D Vehicle Detection with Pseudo Shape Segmentation and Adaptive Camera-LiDAR Fusion. *IEEE Trans. Ind. Inform.* **2023**, 1–11. [CrossRef]
26. Carranza-García, M.; Torres-Mateo, J.; Lara-Benítez, P.; García-Gutiérrez, J. On the Performance of One-Stage and Two-Stage Object Detectors in Autonomous Vehicles Using Camera Data. *Remote Sens.* **2021**, *13*, 89. [CrossRef]
27. Jiang, P.; Ergu, D.; Liu, F.; Cai, Y.; Ma, B. A Review of Yolo Algorithm Developments. *Procedia Comput. Sci.* **2022**, *199*, 1066–1073.
28. Wang, C.Y.; Bochkovskiy, A.; Liao, H.Y.M. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *arXiv* **2022**, arXiv:2207.02696.
29. Dosovitskiy, A.; Ros, G.; Codevilla, F.; Lopez, A.; Koltun, V. CARLA: An open urban driving simulator. In Proceedings of the Conference on Robot Learning (PMLR), Mountain View, CA, USA, 13–15 November 2017; pp. 1–16.
30. Abdulla, W. Mask R-CNN for Object Detection and Instance Segmentation on Keras and TensorFlow. 2017. Available online: https://github.com/matterport/Mask_RCNN (accessed on 15 March 2023).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.