

LiCAR: pseudo-RGB LiDAR image for CAR segmentation

Ignacio de Loyola Páez-Ubieta^[0000-0001-9901-7264], Edison P. Velasco-Sánchez^[0000-0003-2837-2001] and Santiago T. Puente^[0000-0002-6175-600X]

AUtomatics, RObotics, and Artificial Vision Lab, IUII: University Institute for Computer Research University of Alicante, Crta. San Vicente s/n, San Vicente del Raspeig, E-03690, Alicante, Spain {edison.velasco, ignacio.paez, santiago.puente}@ua.es

Abstract. With the advancement of computing resources, an increasing number of Neural Networks (NNs) are appearing for image detection and segmentation appear. However, these methods usually accept as input a RGB 2D image. On the other side, Light Detection And Ranging (LiDAR) sensors with many layers provide images that are similar to those obtained from a traditional low resolution RGB camera. Following this principle, a new dataset for segmenting cars in pseudo-RGB images has been generated. This dataset combines the information given by the LiDAR sensor into a Spherical Range Image (SRI), concretely the reflectivity, near infrared and signal intensity 2D images. These images are then fed into instance segmentation NNs. These NNs segment the cars that appear in these images, having as result a Bounding Box (BB) and mask precision of 88% and 81.5% respectively with You Only Look Once (YOLO)-v8 large. By using this segmentation NN, some trackers have been applied so as to follow each car segmented instance along a video feed, having great performance in real world experiments.

Keywords: Artificial Intelligence · Mobile Robots · Neural Networks · Instance Segmentation · LiDAR · YOLO.

1 Introduction

The first Artificial Intelligence (AI) models date back to 1943 [9]. Initially, the goal was to simulate the function of a single neuron. Since then, several advances have led to the development of more complex structures, the most recent of which are data storage [6] and Graphics Processing Units (GPUs) [3], in 2006 and 2010 respectively.

Regarding object detection and segmentation methods, there are three different approaches; single, two and multi-stage methods [5]. Among them, the first one is the one that receives more attention, since they are usually faster [14], allowing to have inference times close to realtime. Some examples of these single-stage methods are You Only Look Once (YOLO) or Single Shot Detector (SSD) [8].

Normally, these methods require a RGB image as input. However, variations have been introduced in recent years that allow the input of pseudo-RGB images generated from Light Detection And Ranging (LiDAR) sensors. For example, [4] uses a combination of RGB and LiDAR images to perform object detection by first identifying regions of interest in the LiDAR point cloud. Using these regions, an RGB image is extracted from the original RGB image and used as input to a YOLO Neural Network (NN), which finally detects the cars. Although this method gives good results, it requires the handling of LiDAR and RGB images and does not detect objects accurately. Other papers, such as [10], generate a pseudo-RGB image from a LiDAR sensor. This is used to detect and track pedestrians across the image. To do this, they slide the Bounding Box (BB) in 5 parts to build the descriptor. However, it only performs object detection, not segmentation, using an old YOLO distribution, since at the time of submission new YOLO distributions had appeared. Finally, [19] generates a pseudo-RGB image from a LiDAR sensor in a similar way. Using detection and segmentation methods, they achieve good results in both indoor and outdoor environments. However, the need to pre-process the image from 2048*128 to 1000*300 and the small number of images and instances in their dataset question the quality of the results obtained.

In this work, we perform the segmentation of car instances on Spherical Range Images (SRIs), obtained by combining three different channels from a LiDAR sensor, with several single-stage methods from the YOLO family.

The main contributions of this work are:

- A new dataset for segmenting cars in pseudo-RGB 2D images has been generated. These images are SRI obtained by combining several channels from a LiDAR sensor.
- Several State Of The Art (SOTA) NN instance segmentation methods have been trained on the aforementioned dataset, achieving almost 90% in BB detection and more than 80% in mask segmentation.
- Once the best instance segmentation was obtained, a tracker was incorporated, achieving excellent results without affecting real-time inference.

This paper is organised as follows: Section 2 introduces both the image generation process and the segmentation NNs used, Section 3 shows the experimental setup of the system, Section 4 shows the results obtained on the test set and also on completely new real world scenarios and Section 5 summarises the results of the paper, as well as introduces the main lines of future works.

2 Methodology

In this Section, the process used to generate the images and the segmentation NNs that were used in Sec. 3 are presented.

2.1 Image generation

Generating images from a LiDAR sensor involves two steps. In the first one, a pseudo-RGB pointcloud is generated from the data provided by the LiDAR,

and in the second one, the previously generated pointcloud is transformed into a SRI.

For the first process, data is obtained from the LiDAR sensor. The used sensor provides 4 different 2D images: near infrared photons, range information, calculated calibrated reflectivity and signal intensity photons images. From them, a pseudo-RGB point cloud is generated by combining the calculated calibrated reflectance, NIR photon and signal intensity photon images.

For the second one, the generated pointcloud is transformed into a SRI. Since the used LiDAR has enough resolution, no interpolation or filtering process is needed to generate the final SRI [17].

An example of the process of combining the LiDAR channels to produce the final pseudo-RGB SRI is shown on Fig. 1.

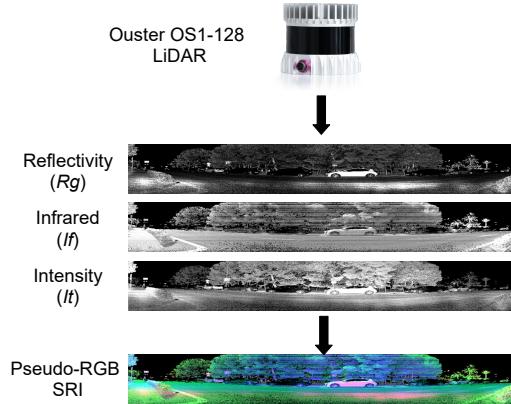


Fig. 1. Combination of LiDAR channels for generating the pseudo-RGB image.

2.2 Segmentation NNs

A fast but accurate method is required to perform instance segmentation. For this purpose, single-stage SOTA methods such as YOLO-v5 [7], YOLO-v7 [18] and YOLO-v8 [16] were chosen.

YOLO-v5 appeared online on 2020. This model was the first in the YOLO NNs family to support instance segmentation, as well as being released by a private company. There is not much hindsight on this model, as no official paper was published. But it seems that the anchor box selection process was built into the model, allowing it to automatically learn the most appropriate size for each dataset [20].

YOLO-v7 appeared online on 2022, but the paper was published in 2023. They introduced a trainable bag-of-freebies, a new reparametrization module, and a composite scaling method. All these features were used to achieve an improvement in the loss function, label assignment, support for multiple architectures, or a good trade-off between network parameters, computation, inference speed, and accuracy.

YOLO-v8 also appeared online on 2022, but later than YOLO-v7, having its paper publication in 2024. To improve on previous versions, a new loss function called focal loss, a new data enhancement method called mixup [21], and a new evaluation metric called Average Precision Across Scale (APAS) were introduced. These allow to focus on complex images instead of simple ones, to improve the generalisation and strength of the model by merging images and labels, and to ensure the accuracy of the model over different scales.

3 Experiments

In this Section, the elements used for the execution of the experiments are presented, as well as details generated dataset. The NNs training procedure and the results obtained are shown in the next section.

3.1 Setup

The Ouster OS1-128 3D-LiDAR was used to obtain the dataset. This sensor is mounted on the roBot for Localization in Unstructured Environments (BLUE) robot, a mobile robotic platform designed by the research group several years ago [12,13], but which is still being updated today [2]. The NNs was trained on a computer with a NVIDIA DGX-A100 Tensor Core GPU with 40GB of RAM. For data acquisition and inference, an on board computer with an AMD Ryzen 7 5700G as Central Processing Unit (CPU), a NVIDIA RTX 3060 with 12 GB as GPU and 32 GB of Random-Access Memory (RAM) was used. The BLUE robot with Ouster OS1-128 3D-LiDAR and on board computer is shown in Fig. 2.

3.2 Dataset

After obtaining the data from the LiDAR sensor and converting it to a SRI, 400 images were obtained, each with approximately 14 car instances. These images contain different outdoor environments from the University of Alicante and have a resolution of 2048x128 pixels. The cars appearing in the images were labelled using the LabelMe [15] tool. These generated labels include a special encoding used to generate YOLO NNs to understand that a partially occluded object belongs to the same instance, so that it is included in the same BB and mask instance.

The generated dataset has been divided into training/validation/test sets according to the 85/10/5 ratio. A summary of the generated dataset can be found in Table 1. This dataset is available online at LICAR dataset.



Fig. 2. BLUE robot with both the Ouster OS1-128 3D-LiDAR and the on board computer.

Table 1. Generated dataset specifications.

| Set | Images | Instances |
|------------|--------|-----------|
| Train | 340 | 4784 |
| Validation | 40 | 569 |
| Test | 20 | 299 |

3.3 NN training

All trainings were performed using transfer learning from previously trained models provided by each NN model, as it has shown to improve model accuracy while training the models in [11].

YOLO-v5 was trained for 300 epochs with a batch size of 128 and 8 workers, using a single channel rectangular image with an image size of 2048 pixels as the longest side. YOLO-v7 was trained for 100 epochs with a batch size of 64 and only 1 worker, also using single channel rectangular images of 2048 pixels. YOLO-v8 was trained for 100 epochs with a batch size of 256, using again the single channel rectangular 2048 pixel rectangular image.

The rest of the used parameters used in each case are the default ones provided by each of the models as recommended parameters for performing training of instance segmentation models.

4 Results

This section presents the results of the experiments described on Section 3.

4.1 Test set

After training the aforementioned models using the training and validation sets, the final results are obtained. The test set contained 20 images with 299 instances, representing all types of data used during the training process. However, in order to have a fair comparison, some parameters of the used NNs are given in Table 2, as well as the computation times per image (executed in the onboard BLUE computer with a NVIDIA RTX 3060 GPU). These computation times are expressed as the sum of three values, which consist of the time spent during preprocess, inference and postprocess steps. So, for example, YOLO-v7 has a similar number of layers to YOLO-v8 medium, but is between YOLO-v8 medium and YOLO-v8 large in terms of the number of parameters. Another example is YOLO-v5, which has fewer layers than YOLO-v8 nano, but is between YOLO-v8 small and YOLO-v8 medium in terms of parameters.

Table 2. Some NN specifications.

| NN | Speed (ms) | Number of layers | Number of parameters |
|----------------|--------------|------------------|----------------------|
| YOLO-v8 nano | 0.3+6.5+0.5 | 261 | 3263811 |
| YOLO-v8 small | 0.3+8.3+0.5 | 261 | 11790483 |
| YOLO-v8 medium | 0.3+18.2+0.6 | 331 | 27240227 |
| YOLO-v8 large | 0.3+28.7+0.5 | 401 | 45912659 |
| YOLO-v7 | 0.9+23.7+0.7 | 325 | 37842476 |
| YOLO-v5 medium | 0.9+15.9+0.7 | 220 | 21652358 |

The results for BB detection and mask segmentation are shown in Table 3 and 4 respectively. In terms of BB detection, YOLO-v8 medium and large have very close results, with the precision being higher for the YOLO-v8 large model. However, when moving to the mask segmentation task, YOLO-v8 would be the model with the best results in 3 out of the 4 metrics evaluated.

Also, all the trained models fall within the real-time specification, as their computation time should be between 33.3 and 66.6 ms for 30 and 15 Frames Per Second (FPS). So our final choice is YOLO-v8 large.

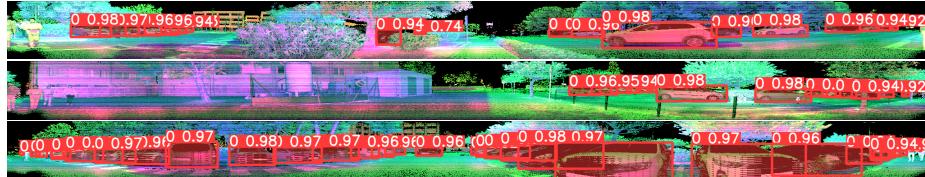
Table 3. Results in BB detection.

| NN | Precision | Recall | mAP @0.5 | mAP@0.5-0.95 |
|----------------|-------------|--------------|--------------|--------------|
| YOLO-v8 nano | 0.858 | 0.756 | 0.85 | 0.589 |
| YOLO-v8 small | 0.836 | 0.832 | 0.871 | 0.633 |
| YOLO-v8 medium | 0.824 | 0.832 | 0.888 | 0.654 |
| YOLO-v8 large | 0.88 | 0.811 | 0.881 | 0.649 |
| YOLO-v7 | 0.81 | 0.823 | 0.87 | 0.62 |
| YOLO-v5 medium | 0.783 | 0.835 | 0.849 | 0.587 |

Table 4. Results in mask segmentation.

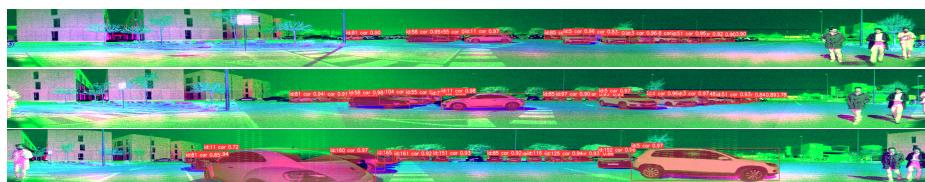
| NN | Precision | Recall | mAP @0.5 | mAP@0.5-0.95 |
|----------------|--------------|--------------|--------------|--------------|
| YOLO-v8 nano | 0.795 | 0.676 | 0.758 | 0.448 |
| YOLO-v8 small | 0.77 | 0.749 | 0.803 | 0.471 |
| YOLO-v8 medium | 0.782 | 0.732 | 0.807 | 0.477 |
| YOLO-v8 large | 0.815 | 0.751 | 0.826 | 0.509 |
| YOLO-v7 | 0.803 | 0.679 | 0.769 | 0.474 |
| YOLO-v5 medium | 0.831 | 0.605 | 0.732 | 0.431 |

Some images of the test subset segmented by YOLO-v8 large can be seen in Fig. 3. As can be seen, cars are correctly segmented in both near and far distance in all images, representing the 360 degrees of the LiDAR sensor Field Of View (FOV).

**Fig. 3.** Some samples from test set segmented by YOLO-v8 large.

4.2 Real experiments

Once the best segmentation method was found, it was tested in a completely new and unseen environment. Also some trackers like BoT-SORT [1] and ByteTrack [22] were added to keep the position of different segmented instances across frames on a video. Some frames of a sample video using the BoT-SORT tracker are shown in Fig. 4. This tracker was chosen over ByteTrack because it represents an evolution, including some improvements such as a camera compensation based feature tracker and a suitable Kalman filter.

**Fig. 4.** Several frames running YOLO-v8 large using BoT-SORT as tracker.

BoT-SORT was used with 0.7 as the threshold for first and second association, 0.75 as the threshold for initialising a new track if the detection does not match any previous track, 20 as the buffer size for removing tracks and 0.8 as the threshold for matching tracks.

5 Conclusion

This paper presents the segmentation of instances in SRI using several single-step methods such as YOLO-v5, YOLO-v6 and YOLO-v8. Specifically, the aim was to detect cars in pseudo-RGB images obtained from a LiDAR sensor by combining several of the different channels available. The results have shown that it is possible to perform instance segmentation in a LiDAR image using YOLO-v8 large, having a precision of 81.5%. This model provided the best accuracy in mask segmentation, allowing real-time inference on BLUE's on-board computer and the mobile robot to segment cars in outdoor environments.

Future work will consist of using the designed system to segment different types of objects, as well as designing a new method for calibrating LiDAR and the camera using the segmented masks obtained with this method.

Acknowledgments. Research work was funded by grant PID2021-122685OB-I00 funded by MICIU/AEI/10.13039/501100011033 and ERDF/EU and grand PRE2019-088069 funded by MICIU/AEI/10.13039/501100011033 and ESF Investing in your future. The computer facilities were provided through the IDIFEFER/2020/003 project.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Aharon, N., Orfaig, R., Bobrovsky, B.Z.: Bot-sort: Robust associations multi-pedestrian tracking. arXiv preprint (2022). <https://doi.org/10.48550/arXiv.2206.14651>
2. Castaño Amorós, J., Páez Ubieta, I.d.L., Muñoz-Bañón, M.Á., Velasco, E.P., Candelas-Herías, F.A., Puente Méndez, S.T., Gil, P., Torres, F.: Desarrollos en blue para manipulación móvil en entornos exteriores no estructurado. In: XLIII Jornadas de Automática (JA). pp. 851–857 (2022). <https://doi.org/10.17979/spudc.9788497498418.0851>
3. Cireşan, D.C., Meier, U., Gambardella, L.M., Schmidhuber, J.: Deep, big, simple neural nets for handwritten digit recognition. Neural computation **22**(12), 3207–3220 (2010). https://doi.org/10.1162/NECO_a_00052
4. Fan, Y.C., Yelamandala, C.M., Chen, T.W., Huang, C.J.: Real-time object detection for lidar based on ls-r-yolov4 neural network. Journal of Sensors **2021**(1), 5576262 (2021). <https://doi.org/10.1155/2021/5576262>
5. Gu, W., Bai, S., Kong, L.: A review on 2d instance segmentation based on deep neural networks. Image and Vision Computing **120**, 104401 (2022). <https://doi.org/10.1016/j.imavis.2022.104401>

6. Hinton, G.E., Osindero, S., Teh, Y.W.: A fast learning algorithm for deep belief nets. *Neural computation* **18**(7), 1527–1554 (2006). <https://doi.org/10.1162/neco.2006.18.7.1527>
7. Jocher, G., Chaurasia, A., Stoken, A., Borovec, J., et al.: ultralytics/yolov5: v6.2 - yolov5 classification models, apple m1, reproducibility, clearml and deci.ai integrations (aug 2022). <https://doi.org/10.5281/zenodo.7002879>
8. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. In: Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14. pp. 21–37. Springer (2016). https://doi.org/10.1007/978-3-319-46448-0_2
9. McCulloch, W.S., Pitts, W.: A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics* **5**, 115–133 (1943). <https://doi.org/10.1007/BF02478259>
10. Olivas, A., Munoz-Banón, M.A., Velasco, E., Torres, F.: Robust single object tracking and following by fusion strategy. In: 2023 International Conference on Informatics in Control, Automation and Robotics (ICINCO). vol. 1, pp. 624–631 (2023). <https://doi.org/10.5220/0012178900003543>
11. Páez-Ubieta, I.d.L., Castaño-Amorós, J., Puente, S.T., Gil, P.: Vision and tactile robotic system to grasp litter in outdoor environments. *Journal of Intelligent & Robotic Systems* **109**(2), 36 (2023). <https://doi.org/10.1007/s10846-023-01930-2>
12. del Pino, I., Cova, S., Contreras, M.Á., Candelas, F.A., Torres, F.: Presenting blue: A robot for localization in unstructured environments. In: 2018 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC). pp. 130–135 (2018). <https://doi.org/10.1109/ICARSC.2018.8374172>
13. del Pino, I., Muñoz-Bañón, M.Á., Cova-Rocamora, S., Contreras, M.Á., Candelas, F.A., Torres, F.: Deeper in blue: Development of a robot for localization in unstructured environments. *Journal of Intelligent & Robotic Systems* **98**, 207–225 (2020). <https://doi.org/10.1007/s10846-019-00983-6>
14. Sultana, F., Sufian, A., Dutta, P.: A review of object detection models based on convolutional neural network. *Intelligent computing: image processing based applications* pp. 1–16 (2020). https://doi.org/10.1007/978-981-15-4288-6_1
15. Torralba, A., Russell, B.C., Yuen, J.: Labelme: Online image annotation and applications. *Proceedings of the IEEE* **98**(8), 1467–1484 (2010). <https://doi.org/10.1109/JPROC.2010.2050290>
16. Varghese, R., M., S.: Yolov8: A novel object detection algorithm with enhanced performance and robustness. In: 2024 International Conference on Advances in Data Engineering and Intelligent Computing Systems (ADICS). pp. 1–6 (2024). <https://doi.org/10.1109/ADICS58448.2024.10533619>
17. Velasco-Sánchez, E.P., Muñoz-Bañón, M.Á., Candelas, F.A., Puente, S.T., Torres, F.: Lilo: Lightweight and low-bias lidar odometry method based on spherical range image filtering. *arXiv preprint* (2023). <https://doi.org/10.48550/arXiv.2311.07291>
18. 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. In: 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 7464–7475 (2023). <https://doi.org/10.1109/CVPR52729.2023.00721>
19. Yu, X., Salimpour, S., Queralta, J.P., Westerlund, T.: General-purpose deep learning detection and segmentation models for images from a lidar-based camera sensor. *Sensors* **23**(6), 2936 (2023). <https://doi.org/10.3390/s23062936>
20. Zaidi, S.S.A., Ansari, M.S., Aslam, A., Kanwal, N., Asghar, M., Lee, B.: A survey of modern deep learning based object detection models. *Digital Signal Processing* **126**, 103514 (2022). <https://doi.org/10.1016/j.dsp.2022.103514>

21. Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D.: mixup: Beyond empirical risk minimization. In: Sixth International Conference on Learning Representations (ICLR) (2018). <https://doi.org/10.48550/arXiv.1710.09412>
22. Zhang, Y., Sun, P., Jiang, Y., Yu, D., Weng, F., Yuan, Z., Luo, P., Liu, W., Wang, X.: Bytetrack: Multi-object tracking by associating every detection box. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 1–21. Springer Nature Switzerland (2022). https://doi.org/10.1007/978-3-031-20047-2_1