360º Real-Time 3D Multi-Object Detection and Tracking for Autonomous Vehicle Navigation

Javier del Egido, Carlos Gómez-Huélamo, Luis M. Bergasa, Rafael Barea, Elena López-Guillén, Javier Araluce, Rodrigo Gutiérrez, and Miguel Antunes

Electronics Department, University of Alcalá (UAH), Spain,
javier.egido@uah.es, cram3r95@gmail.com, {luism.bergasa, rafael.barea, elena.lopezg, javier.araluce, miguel.antunes}@uah.es, res.rodry@gmail.com
https://www.robesafe.uah.es/

Abstract. This paper presents a real-time 3D Multi Object Detection and Tracking (DAMOT) method proposed for the UAH autonomous electric car. It allows the vehicle to recognize 360 degrees surrounding objects and uniquely identify them to be able to follow their trajectory in scene by only receiving a 3D point cloud through ROS framework. First, we describe our proposal of 3D object detector, based on PointPillars [1], processing LiDAR data to locate objects in space obtaining their dimensions and location. Secondly, we use BEV-MOT [2], our Multi-Object Tracking technique in order to uniquely identify each object over a Bird's-Eye View (BEV) through a combination of 2D Kalman filter and Hungarian algorithm, allowing the ego-vehicle to follow surrounding objects trajectories. A comparison of the performance of our proposal with other state-of-the-art methods is carried out applying KITTI-3DMOT evaluation tool extracted from AB3DMOT [3] on KITTI [4] validation dataset. Finally, we validate our DAMOT in several traffic scenarios implemented in CARLA [5] open-source driving simulator by using AB4COGT tool, designed by authors, studying its performance in a controlled but realistic urban environment with real-time execution, providing several demonstration videos[1].

Keywords: Real-Time, CARLA, LiDAR, 3D Multi-Object Tracking, ROS, DAMOT, Autonomous Navigation

1 INTRODUCTION

Autonomous driving systems have to perform safe driving behaviours following conventional traffic rules to achieve a programmed destination. When travelling through this route many unforeseen objects may force the controller to react to avoid fatalities. The reliability of the crash-avoidance system relies on the performance of the environment detector and its ability to predict future situations. Detection and Multi-object Tracking (DAMOT) methods analyze an object since its first detection to determine its future position, allowing the car

[1] https://cutt.ly/3rU113d

to act in consequence to avoid critical situations. The increase in object detection performance has allowed to advance on MOT techniques, improving its accuracy but carrying a high computational cost, making prohibitive its use in real-time systems. Many different technologies have been designed to accomplish an optimal environment detector following different approaches, based on different benchmarks such as KITTI [4], which provides manual labeled data from urban scenes taken from different cameras and LiDAR mounted on a vehicle.

Our proposed method applies PointPillars [1], a state-of-the-art 3D object detector network with a striking performance, achieving accurate predictions at a high frame rate taking advantage on GPU and 2D convolutions to process a 3D point cloud.

Our tracking module is formed by our BEV-MOT proposal [2]. It works over a BEV and is composed by a 2D Kalman filter to predict vehicles position between detection frames and Hungarian algorithm in order to associate predictions and detections, running traditional tracking techniques at high speed without sacrificing precision.

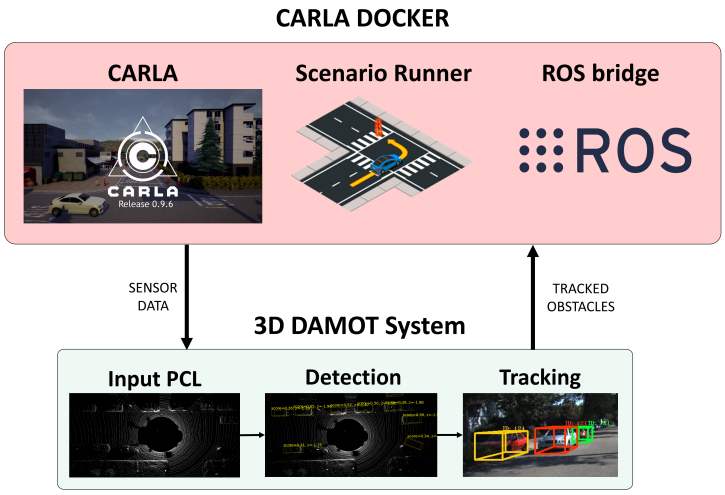


Fig. 1: Architecture of our MOT proposal simulated in CARLA.

We propose to take advantage of the good processing ratios of PointPillars 3D objects detector and traditional tracking techniques to implement a real-time 3D DAMOT able to be used in challenging urban traffic scenarios. The architecture of our proposal integrated to CARLA driving simulator is shown in Fig. 1, which receives a point cloud from ego_vehicle and returns the tracked objects around it at its current and future positions.

2 Related Works

2.1 Vision-Only 3D Object Detection

3D scene understanding through RGB or RGB-D images is possible by processing the information using Convolutional Neural Networks followed by applying complex projection transforms as in [6], [7]. GPUs may be used to reduce computational cost, getting well processing time values. Minor pixel mistakes in image detection may produce meaningful differences between estimated and reality when the object is at a relevant distance, and detecting the object orientation and depth is a difficult task.

2.2 3D LiDAR Object Detection

3D LiDAR sensors provide accurate 3D information, avoiding precision problems when positioning the objects in the three-dimensional space. In contrast to vision-only detection, point clouds provided by LiDAR sensors are independent from adverse weather or light conditions. LiDAR information must be precisely enough to recognize the surroundings in the point cloud but it may not exceed the computational limits due to a excessive detail. Different ways has been explored to develop object recognition using 3D information, such as LiDAR clustering, reducing the complexity to 2D BEV [8], dividing the scene into 3D voxels to apply 2D CNNs as PointPillars[1] or 3D CNNs [9], [10], or processing the whole point cloud [11].

2.3 2D Multi-Object Tracking

2D tracking may be carried out on BEV or image frame. Despite of useful results achieved by batch MOT works like [12], future information is not available when applying tracking to real-time systems such as autonomous vehicles. Otherwise, online 2D MOT traditional techniques make data association using Hungarian Algorithms [13], [14], [15], and more recently Convolutional Networks to extract object characteristics to avoid identity switching when crossing trajectories [16]. Bird's eye view perspective prevents tracking system to experience object occlusions, allowing for a better scene comprehension.

2.4 3D Multi-Object Tracking

3D MOT allows the system to estimate real velocity, undistorted by bi-dimensional transformations, being able to perform predictions based on linear and angular velocities and precise location and size information. The simplest but efficient way to create 3D MOT is to extend 2D Kalman filter by adding a third dimension such as in AB3DMOT [3]. Other works apply simple 2D Kalman filter over BEV and recover the height using different strategies [2], [17].

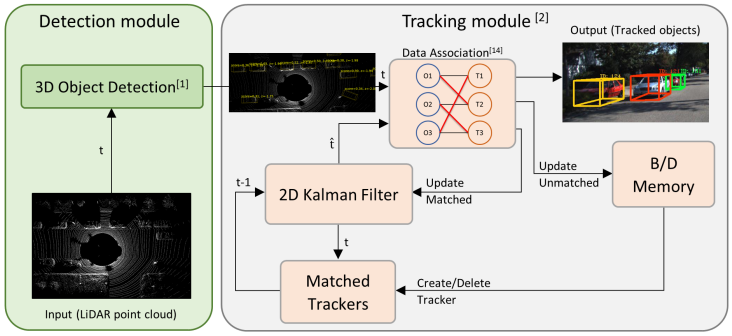


Fig. 2: 3D DAMOT Proposed architecture.

3 Architecture

Our 3D DAMOT system is featured by a modular architecture where first of all a 360º object detector by using a LiDAR sensor is applied. Following, a Kalman Filter and Hungarian Algorithm based tracking approach over the BEV detected objects is carried out. The Fig. 2 outlines the pipeline of our system. The input of the 3D DAMOT consist of a LiDAR point cloud which feeds PointPillars [1] neural network, giving rise to the 3D bounding boxes of the most relevant obstacles of the scene detected in the 360º surrounding of the vehicle. These detections are projected to the road plane (BEV) and analyzed along with our BEV-MOT [2], composed by a 2D Kalman filter, which predicts positions to associate the detected items to previous trackers. Matched trackers update Kalman filter while unmatched objects are processed by birth and death (B/D) module to create new trackers for unmatched objects and delete old trackers for the unmatched ones. All trackers make up the Kalman filter, that estimates the future position of the detected objects to restart the cycle.

3.1 3D object detector

360 degrees surrounding objects are detected processing LiDAR point cloud data with PointPillars[1] network. This end-to-end network analyzes the point cloud by dividing it into vertical columns to process the data as a 2D pseudo-image to take advantage of highly efficient 2D convolutions on GPU, avoiding hand tuning parameters by learning by itself instead. It is trained using KITTI [4] tracking dataset which provides 3D hand-labeled information of vehicles, pedestrians and cyclists in urban environments.

In order to increase the performance of the object detector, several network architectures have been trained for PointPillars, showing their results on the ablation study carried out in 4.2.

3.2 Multi-Object Tracking

Detected objects in a BEV have to be processed by our Multi-Object Tracking module (BEV-MOT) in order to assign a unique identifier to distinguish it along the covered path. Analyzing the objects contained in a scene is a difficult task and has to be accomplished by dividing it into several modules, as seen in Fig. 2.

2D Kalman Filter. By using the constant velocity model, the future position of each tracked object is predicted in the BEV. Kalman filter is updated when present frame objects are matched with current trackers or new ones are created at each step. It also corrects the detected orientation to match it to the assumed previously to avoid a 3D IoU reduction produced by flipping 180 degrees the orientation of the object. To get 3D IoU, height is recovered from the detection module. Different future positions are predicted to calculate possible collisions.

Hungarian Algorithm for Data Association. Detected objects in current frame and predicted positions from past frame are matched applying the Hungarian algorithm according to 3D IoU, recovering height information from 3D object detector, returning matched and unmatched objects and trajectories to be managed by 2D Kalman filter and Birth and Death memory respectively.

Birth and Death Memory. B/D module handles unmatched trackers and objects. When an unmatched object is detected several times (because it newly appears on scene or it cannot be associated with a previously detected one) exceeding F_{min} threshold a new tracker is created. On the other hand, an existing tracker is deleted (due to the object has abandoned the scene or it is associated with a new one) when passed Age_{max} frames.

Matched Trackers. Trackers module manages all current trackers to correctly configure the Kalman filter according to the objects detected in the previous frame.

4 Evaluation

A previous performance evaluation has to be accomplished before to implement a system in a real prototype. To carry out this evaluation on 3D DAMOT metrics two different validation ways have been followed: KITTI real and CARLA simulated scenes.

4.1 3D DAMOT Evaluation Metrics

Traditionally DAMOT evaluation tools study the working behavior on bi-dimensional plane, designed for 2D DAMOT on images. In order to know the performance of our architecture we applied 3D DAMOT benchmark developed by AB3DMOT [3] to analyze 3D IoU from detected and tracked objects with KITTI and CARLA groundtruth. This tool applies traditional and integral metrics described following.

Traditional DAMOT metrics. Mainstream metrics applied to multi-object tracking systems are extracted from CLEAR[18] DAMOT metrics like MOTA (accuracy), MOTP (precision), Mostly-tracked trajectories, False-Negatives (FN), False-Positives (FP), Precision, F1 score, Identity Switching (IDS), Fragmentations (FRAG, trajectory interruptions by false positives), and moreover. This metrics analyze the performance of the DAMOT system at a specific score threshold, not taking into account the confidence provided by the object detector and possibly misunderstanding the capability of the method.

MOTP measures the precision by analyzing the total error in estimated position for each object and its ground-truth.

$$MOTP = \frac{d}{num_{gt}} \tag{1}$$

MOTA considers all error made by the system, false positives, false negatives and identity switching.

$$MOTA = \frac{FP + FN + IDS}{num_{gt}} \tag{2}$$

where num_{gt} is the number of ground truth objects in all frames.

Integral DAMOT metrics. To solve the problem presented by traditional DAMOT metrics, AB3DMOT[3] propose two integral metrics, evaluating the MOTA and MOTP of the tracking system at every score threshold.

$$AMOTA = \frac{1}{L} \sum_{\{\frac{1}{L}, \frac{2}{L}, \ldots, 1\}} (1 - \frac{FP + FN + IDS}{num_{gt}}) \tag{3}$$

where L is the number of different recall values. FP, FN and IDS in AMOTA are modified for each threshold value.

Similarly, AMOTP will be estimated by integrating MOTP along all recall values.

4.2 Object Detector Evaluation in KITTI

Firstly, we modify the 3D object detector by modifying both PointPillars [1] architecture and training steps and keeping the rest of the DAMOT architecture unchanged. All training phases are accomplished by using KITTI [4] database.

In order to compare the results obtained with KITTI-3DMOT evaluation tool we show the performance accomplished by FANTrack[19], which is given in [3]. 3D DAMOT Evaluation parameters are shown in Table 1. To evaluate PointPillars[1] architectures we extracted them from Second.Pytorch GitHub, testing their results for given network weights, which are displayed as PP_lite and PP_onestage, only improving MOTP results.

With the aim of enhance the performance we trained the PP architecture during both 593,920 and 1,187,840 steps by using KITTI[4] tracking dataset for car object class. Table 2 shows the results obtained by the explained detectors,

Table 1: 3D DAMOT evaluation tool default parameters

Parameter	Value
Age_{max}	3
F_{min}	1
IoU	0.1

writing the best result in black and the second in blue, using BEV-MOT [2] as tracking module. It can be observed that PointPillars[1] architecture after 1,187,840 steps (PP Full steps) has the second best performance in 4 of 8 evaluated parameters, improving the overall performance of other PP configurations but no overcoming the [19] output. However, PP configurations run at least two times faster, which is a critical factor in a real-time implementation.

Table 2: Ablation study replacing 3D Object Detector in KITTI-3DMOT evaluation tool in KITTI tracking validation dataset with BEV-MOT [2] tracking.

Method	AMOTA	AMOTP	MOTA	MOTP	IDS	FRAG	FP	FN
[20] Detector	**31.37**	**64.29**	**62.38**	68.26	**1**	**24**	**1215**	**1894**
PP Mid steps	26.14	63.41	57.54	75.12	122	182	4746	**5353**
PP_lite	9.27	56.69	21.54	**79.19**	31	**40**	5635	13219
PP_onestage	10.25	51.51	23.68	**77.78**	**12**	992	7881	10478
PP Full steps	**27.30**	**63.45**	**59.26**	75.24	131	186	**4310**	5364

4.3 CARLA: A Realistic Urban Environment Simulator

Despite impressive efforts made by AB3DMOT [3], where a tool for evaluating 3D DAMOT systems directly in 3D space is designed, it is based on KITTI dataset [4], which provides prerecorded sequences over which the user cannot interact with the environment. Moreover, these sequences are usually based on common driving scenarios, such as a daily quiet street or a highway in which no challenging traffic scenarios as pedestrian crossing, give way, *etc.* takes place.

Regarding levels of automation, neither any research nor industry organization has demonstrated a ratified testing methodology for L4/L5 (being identified the level 5 with a fully-autonomous navigation architecture, according to J3016 SAE document [21]) autonomous vehicles. The reason is quite simple: even though some regulations have been defined for these L4/L5 levels, simulation is a critical aspect to build safe autonomous vehicles. Nevertheless, in spite of the fact that current automotive companies are very good at testing the individual

components of the navigation architecture, these tests are not powerful enough to validate a fully-autonomous navigation architecture on the road, so there is a need to figure out how to test intelligent vehicles full of advanced sensors and sharing information among them [22].

In terms of 3D Multi-Object Detection and Tracking, the answer is quite similar. Since the urban environment is highly complex, the whole architecture and particularly the 3D DAMOT system must be tested in countless traffic scenarios and environments, which would escalate the development time and cost exponentially with the physical approach, either testing at the real-wold or waiting for using new sequences of KITTI (recorded by a physical system), not studying the global advantages and drawbacks of the DAMOT system. For that reason, virtual testing (simulation) and an appropriate design of the traffic scenarios are the keys to build robust and safe autonomous vehicles in the future, as shown in [23]. Since the proposed 3D DAMOT architecture of this work is open-source, we decided to validate the ability to detect and track the most relevant objects around the vehicle in CARLA [5], an open-source hyper-realistic autonomous driving simulator that offers an outstanding environment in terms of perception, flexibility, traffic situations and real-time, which are key concepts for our system.

CARLA is an open-source autonomous driving simulator implemented as a layer over Unreal Engine 4 (UE4) [24]. This simulation engine provides to CARLA an ecosystem of interoperable plugins, a realistic physics and a state-of-the-art image quality. CARLA is designed as a server-client system so as to support this functionality provided by UE4, where the simulation is rendered and run by the server. The environment is composed of 3D models of static objects, such as buildings, infrastructure or vegetation, as well as dynamic objects like pedestrians, cyclists or vehicles. These objects are designed using low-weight geometric textures and models though maintaining visual realism by making use of variable level of detail and carefully crafting the materials. Moreover, one of the main advantages when using CARLA is the possibility to modify in an easy way the vehicle on-board sensors and their features in order to obtain accurate data, the weather and even the possibility to create realistic traffic scenarios as in Fig. 3.

In order to obtain the point cloud required by our 3D object detector, we use the CARLA ROS bridge, associated to the CARLA simulator. The CARLA ROS bridge is a ROS package that aims at providing a bridge between CARLA and ROS (Robot Operating System [25]), sending the information captured by the on-board sensors and other variables of interest to the vehicle in the form of parameters and topics understood by ROS. In this paper we used the 0.9.9 version of CARLA in such a way the ROS bridge was configured according to this version. In terms of the sensors perspective, the agent sensor suite can be modified in a flexible way. Most common sensors in CARLA world are GPS, RGB cameras and LiDAR (in addition to their corresponding pseudo-sensors, such us the semantic segmentation and the ground-truth associated to the RGB information of a camera). Since in this work our object detector is only LiDAR

Fig. 3: CARLA driving simulator

based, we configure the sensor as shown in Table 3. Based on the bridge, the 3D point cloud captured by the LiDAR is published in ROS format as PointCloud2, with the X axe inwards, Y left and Z pointing up. Also an RGB camera sensor is defined backwards the car to show the performance of the 360º 3D DAMOT system to detect and track the detected objects all around the vehicle as seen in Fig. 8b, but not interfering with the operation of our system.

Table 3: LiDAR configuration in CARLA simulator

Parameter	Value
X (m)	0.0
Y (m)	0.0
Z (m)	2.5
Points per second	500.000
Upper FoV (º)	2.0
Lower FoV (º)	-15.0
Rotation frequency (Hz)	20

As shown, CARLA provides a straightforward way to add or remove sensors from the vehicle or even modify their parameters, to adjust the simulation to the real-world as best as possible.

One of the best advantages of CARLA is the possibility to create ad-hoc urban layouts, helpful to validate the 3D DAMOT system under different traffic and weather conditions. CARLA Scenario Runner module can be downloaded from CARLA GitHub, obtaining an execution engine for CARLA and traffic scenario definition. These scenarios can be modified by editing an OpenSCENARIO [26] script definition where town, vehicles, climate conditions and also

driving behaviours are defined. Fig. 3 depicts an scenario on Town10 with the ego_vehicle and a predefined route, showing waypoints on a curved street to achieve the destination point.

As previously explained, DAMOT evaluation in KITTI is mainly carried out along daily streets where many cars are parked on the road, so it mostly evaluate the system performance to track static vehicles when the main difficult is found in dynamic obstacles, such as pedestrians, vans, trucks or cars. To achieve more challenging situations we validated our DAMOT method on different CARLA simulated scenarios.

4.4 DAMOT Evaluation in CARLA

Figure 4 illustrates the followed process, validating both DAMOT architectures based on our BEV-MOT [2] and AB3DMOT [3] (our baseline) on CARLA simulator using AB4COGT, a specific tool designed by authors to extract CARLA objects information used as ground-truth on KITTI-3DMOT evaluation tool developed by AB3DMOT.

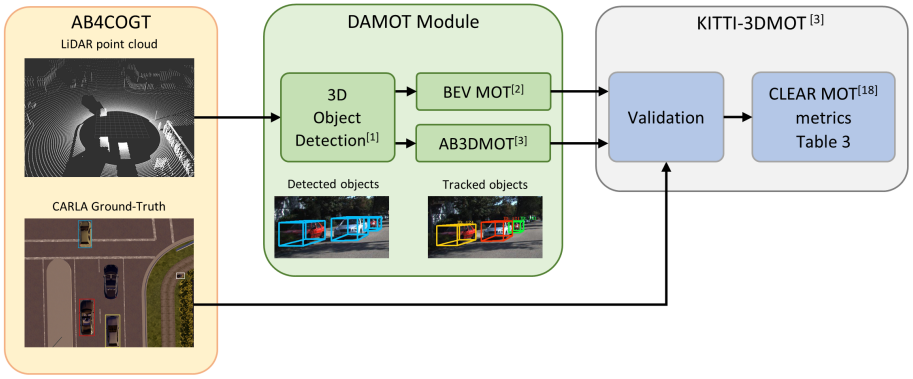


Fig. 4: DAMOT validation through AB4COGT and KITTI-3DMOT

To demonstrate the potential of validation on simulated environments by using AB4COGT, several scenarios are defined in CARLA driving simulator. The ego_vehicle used in all scenarios mounts a top LiDAR configured as described in Table 3. The scenes are saved using AB4COGT tool, recording the object when it is around the ego-vehicle and within a 50 meters radius, storing both ground-truth labels and LiDAR point clouds to ensure synchronization. Then, point clouds are reproduced as ROS message to stimulate PointPillars [1] input, detecting the object in the scene. Tracking is then applied to obtain CLEAR MOT [18] metrics.

Scenario 1. Parked aside vehicles. In this scenario we replicated a KITTI-like scene to demonstrate the similarity between simulated and real scenes. The

autonomous ego_vehicle travels along a street where several vehicles are parked aside, reproducing an environment similar to KITTI traffic scenes, without difficulties to complete the planned path.

Figure 5 shows the disposal of the objects in scene, displaying a BEV of CARLA simulator scene, a BEV of detected and tracked vehicles, and a perspective view of LiDAR point cloud and tracked objects to verify correspondence.

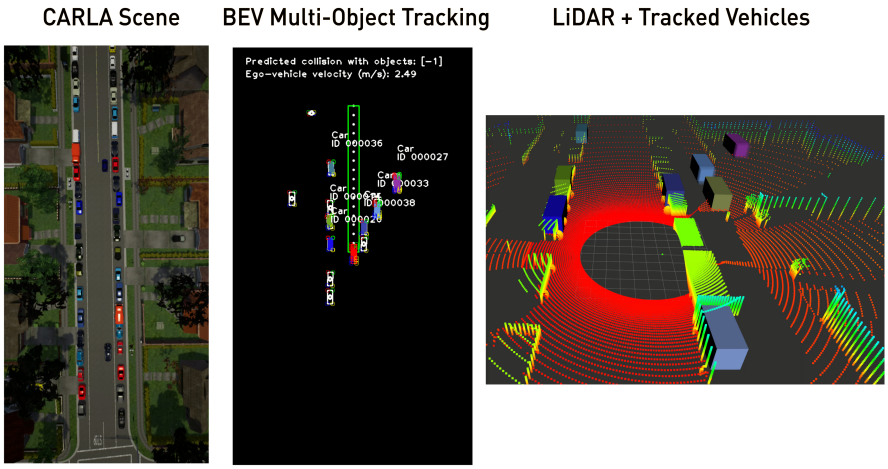
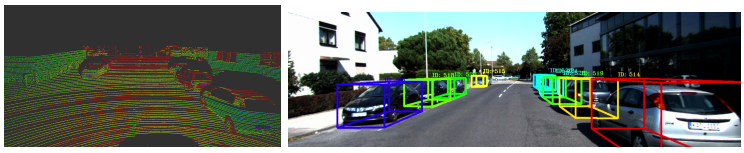


Fig. 5: Scenario 1. Parked aside vehicles.

It can be observed in Fig. 5 that CARLA provided LiDAR point cloud is not accurate enough to vehicles shape, decreasing the ability of PointPillars to detect cars due to its training on real scenarios, with accurate vehicles shape. A similar scene provided by KITTI is displayed in Fig 6. In this case, a real LiDAR point cloud is given to PointPillars input, being able to detect much more vehicles and consequently producing a better and more stable objects tracking.



(a) KITTI point cloud. (b) Detected and tracked vehicles.

Fig. 6: KITTI scene DAMOT and point cloud.

Scenario 2. Stopped at junction. The second scenario takes place at a simulated traffic-light regulated junction, where the ego_vehicle detects and tracks moving vehicles, introducing detection and tracking on dynamic objects. It can be observed on Fig 7 that, despite of time delays shown on tracked vehicle markers, the vehicles are correctly detected and uniquely identified, showing a fuzzy trail on predicted future positions.

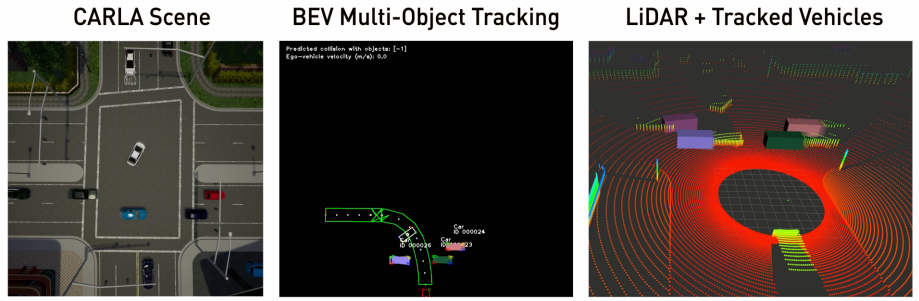
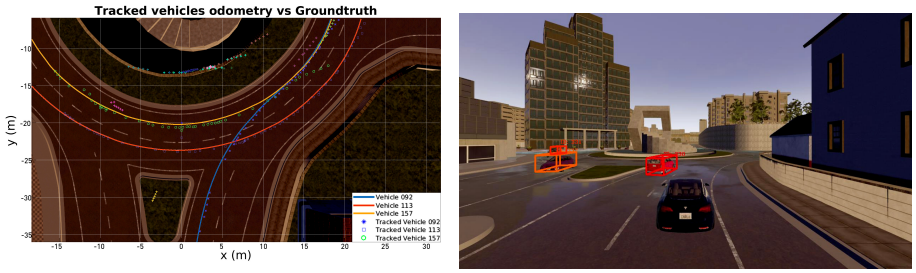


Fig. 7: Scenario 2. Stopped at traffic-regulated junction.

Scenario 3. Entering a roundabout. Last scenario recreate a situation considering the opposite condition of KITTI, that is, our ego-vehicle, represented by a Tesla Model 3, is static aside of the road whilst the other vehicles are moving around our vehicle. In order to validate the performance of our DAMOT system, we store the odometry of the vehicles obtained through the odometry ROS topics defined for each vehicle (ground-truth), while their predicted position and associated identifier are obtained from the MOT system.



(a) Tracked odometry vs ground-truth. (b) RGB view.

Fig. 8: Scenario 3. Entering a roundabout.

In Fig. 8a vehicles ground-truth trajectory are drawn as a continuous line while tracked vehicles are shown as asterisks, squares and circumferences re-

spectively and other detected objects appears as crosses. It can be appreciated how close are predicted and ground-truth positions for each vehicle, and the consistency of the tracker, which keeps a correct association to each vehicle for almost all the scene. Also it can be noted that most false positives are due to roundabout urban decoration elements.

Fig. 8b shows main ego_vehicle centered on image and projected 3D bounding boxes overlapping the tracked vehicles. RGB camera is not needed for tracking and is only used for demonstrative use.

DAMOT validation results. Table 4 shows the results of the validation tests for the above scenarios using the KITTI-3DMOT tool based on validation code developed by [3], comparing two tracking methods, providing a methodology to test this kind of systems in every traffic condition without taking any risk.

Table 4: DAMOT validation on CARLA scenarios.

Method	AMOTA	AMOTP	MOTA	MOTP	IDS	FRAG	FP	FN
AB3DMOT [3]	21.92	**63.67**	32.60	**69.27**	0	**26**	68	**178**
BEV MOT [2]	**26.45**	12.04	**48.77**	22.09	0	28	**0**	187

It can be seen that our BEV-MOT [2] tracking method achieves better AMOTA and MOTA results than AB3DMOT[3], which are intrinsically related with tracking metrics, obtaining a lower result on AMOTP and MOTP metrics, associated with a worst detection. According to these results, BEV-MOT tracking method is more efficient and demonstrates a better performance than AB3DMOT, increasing AMOTP and MOTP values when obtaining a better detection in a more realistic LiDAR point cloud.

5 CONCLUSIONS

This work presents the architecture and validation of our 360 degrees Real-Time 3D Multi-Object Detection and Tracking system on KITTI dataset with 3D DAMOT evaluation tool and CARLA simulator, a flexible and hyper-realistic open-source simulator for autonomous driving. The implemented method allows to track static and dynamic objects around an autonomous car in real-time to enhance its safety system by only using a 3D LiDAR point cloud as input. The validation has consisted of firstly analyze the results on KITTI validation set modifying the 3D object detector architecture and the trained weights. After that, different traffic scenarios has been implemented in CARLA to study the performance of our DAMOT architecture in a realistic urban environment, comparing our BEV-MOT proposal with AB3DMOT (our baseline). As future works,

the system will be mounted on our real electric autonomous car to fully test its ability to detect and track vehicles and pedestrians in traffic urban scenarios, improving the robustness of our current tracking system. Also different sensors such as radars and cameras will be added in order to strengthen the DAMOT module behaviour. Furthermore, we hope CARLA will improve LiDAR point cloud sensor in order to produce more realistic vehicles shapes to approximate PointPillars ability to detect object to real world datasets.

ACKNOWLEDGMENT

This work has been funded in part from the Spanish MICINN/FEDER through the Techs4AgeCar project (RTI2018-099263-B-C21)and from the RoboCity2030-DIH-CM project (P2018/NMT- 4331)), funded by Programas de actividades I+D (CAM) and cofunded by EU Structural Funds.

References

1. A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2019.
2. C. Gómez-Huelamo, J. del Egido, L. M. Bergasa, R. Barea, M. Ocaña, F. Arango, and R. Gútierrez, "Real-time bird's eye view multi-object tracking system based on fast encoders for object detection," in *2020 IEEE Intelligent Transportation Systems Conference (ITSC)*, IEEE, 2020.
3. X. Weng and K. Kitani, "A baseline for 3d multi-object tracking," 2019.
4. 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*, pp. 3354–3361, IEEE, 2012.
5. A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," 2017.
6. Z. Qin, J. Wang, and Y. Lu, "Triangulation learning network: From monocular to stereo 3d object detection," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.
7. H. Konigshof, N. Salscheider, and C. Stiller, "Realtime 3d object detection for automated driving using stereo vision and semantic information," pp. 1405–1410, 10 2019.
8. B. Yang, M. Liang, and R. Urtasun, "Hdnet: Exploiting hd maps for 3d object detection," in *Proceedings of The 2nd Conference on Robot Learning* (A. Billard, A. Dragan, J. Peters, and J. Morimoto, eds.), vol. 87 of *Proceedings of Machine Learning Research*, pp. 146–155, PMLR, 29–31 Oct 2018.
9. W. Luo, B. Yang, and R. Urtasun, "Fast and furious: Real time end-to-end 3d detection, tracking and motion forecasting with a single convolutional net," pp. 3569–3577, 06 2018.
10. Y. Zhou and O. Tuzel, "Voxelnet: End-to-end learning for point cloud based 3d object detection," pp. 4490–4499, 06 2018.
11. S. Shi, X. Wang, and H. Li, "Pointrcnn: 3d object proposal generation and detection from point cloud," 2018.

12. S. Schulter, P. Vernaza, W. Choi, and M. Chandraker, "Deep network flow for multi-object tracking," 2017.
13. N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," 2017.
14. H. W. Kuhn and B. Yaw, "The hungarian method for the assignment problem," *Naval Res. Logist. Quart*, pp. 83–97, 1955.
15. A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking," 2016.
16. N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," 2017.
17. A. Patil, S. Malla, H. Gang, and Y.-T. Chen, "The h3d dataset for full-surround 3d multi-object detection and tracking in crowded urban scenes," 2019.
18. K. Bernardin and R. Stiefelhagen, "Evaluating multiple object tracking performance: The clear mot metrics," *EURASIP Journal on Image and Video Processing*, vol. 2008, 01 2008.
19. E. Baser, V. Balasubramanian, P. Bhattacharyya, and K. Czarnecki, "Fantrack: 3d multi-object tracking with feature association network," 2019.
20. X. Weng and K. Kitani, "Monocular 3d object detection with pseudo-lidar point cloud," 03 2019.
21. S. Taxonomy, "Definitions for terms related to driving automation systems for on-road motor vehicles (j3016)," tech. rep., Technical report, Society for Automotive Engineering, 2016.
22. H. Schöner, "The role of simulation in development and testing of autonomous vehicles," in *Driving Simulation Conference, Stuttgart*, 2017.
23. C. Gómez-Huelamo, L. M. Bergasa, R. Barea, E. López-Guillén, F. Arango, and P. Sánchez, "Simulating use cases for the uah autonomous electric car," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pp. 2305–2311, IEEE, 2019.
24. A. Sanders, *An introduction to unreal engine 4*. AK Peters/CRC Press, 2016.
25. M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Ng, "Ros: an open-source robot operating system," vol. 3, 01 2009.
26. J.-M. Jullien, C. Martel, L. Vignollet, and M. Wentland, "Openscenario: a flexible integrated environment to develop educational activities based on pedagogical scenarios," in *2009 Ninth IEEE International Conference on Advanced Learning Technologies*, pp. 509–513, IEEE, 2009.