Recognize Moving Objects Around an Autonomous Vehicle Considering a Deep-learning Detector Model and Dynamic Bayesian Occupancy

Andrés Eduardo Gómez Hernandez, Özgür Erkent, Christian Laugier

Recognize Moving Objects Around an Autonomous Vehicle Considering a Deep-learning Detector Model and Dynamic Bayesian Occupancy

Andrés E. Gómez Hernandez[1], Özgür Erkent[1] and Christian Laugier[1]

***Abstract*— Perception systems on autonomous vehicles have the challenge of understanding the traffic scene in different situations. The fusion of redundant information obtained from different sources has been shown considerable progress under different methodologies to achieve this objective. However, new opportunities are available to obtain better fusion results with the advance of deep-learning models and computing hardware. In this paper, we aim to recognize moving objects in traffic scenes through the fusion of semantic information with occupancy-grid estimations. Our approach considers a deep-learning model with inference times between 22 to 55 milliseconds. Moreover, we use a Bayesian occupancy framework with a Highly-parallelized design to obtain the occupancy-grid estimations.We validate our approach using experimental results with real-world data on urban scenery.**

I. INTRODUCTION

The understanding of a traffic scene is a constant challenge for research in autonomous vehicles. The development of autonomous vehicles has to consider several factors like weather conditions, obstacles, traffic flow, among others[1]. Regarding the obstacles in the environment, their detection, recognition, and dynamic estimation are still a significant issue in this area.

In the last years, a considerable number of approaches for object detection and motion estimation have considered obtaining information from a single sensor. RGB cameras are the best example. Several algorithms in state of the art use this sensor given the wide variety of information on the scene traffic, affordability, and availability[2], [3]. [4], [5], [6], [7] are some proposals for the detection and recognition of objects in the scene that only used RGB cameras. However, RGB cameras are susceptible to light and weather conditions. Moreover, monocular cameras cannot obtain depth information from the objects in the scene [8].

Other proposals used the lidar sensor for object detection and motion estimation [9], [10], [11]. The lidar sensor permit to obtain a field of view of 360 degrees with a precise distance measurement from the obstacles in the scene. Furthermore, object detection using the lidar has not problems with adverse lighting conditions. However, considering the technical characteristics of the lidar, the 3D points result from the sensor reading do not provide optimal texture information. Finally, the price of the lidar sensor is its main disadvantage. New solid-state technologies could be an alternative to decrease the current lidar price[8].

[1] Authors are with Univ. Grenoble Alpes, INRIA, Chroma Team, France INRIA, Rhône-Alpes, France. Correspondence: andres.gomez-hernandez@inria.fr

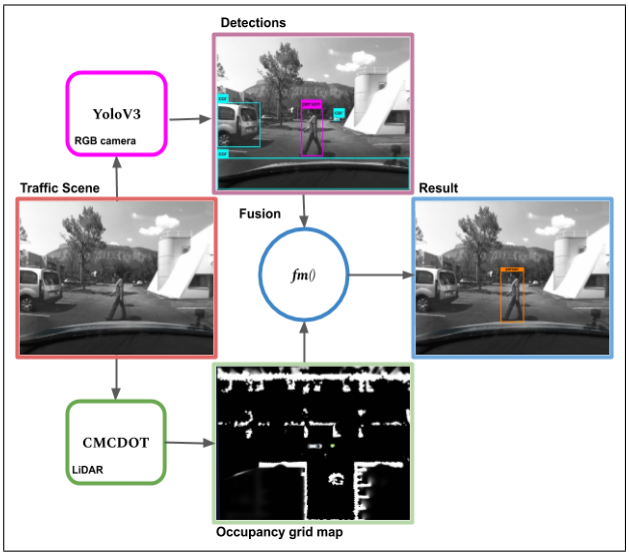

Fig. 1: Fusion model description. Figure shows the two approaches(YoloV3 and CMCDOT framework) used to interpret the scene.

Another trend to resolve the problem of recognizing dynamic objects in the scene could be the development of fusion models [12]. These models contemplate the advantages and disadvantages of different sensing modalities, such as camera and lidar. The fusion models also consider the synchronous acquisition of information preprocessed provided by sensors [13]. Therefore, the fusion models have to be efficient and fast to decode data time series from different sources.

In this paper, we propose a new approach to recognize moving objects around an autonomous vehicle by fusing preprocessed information from two sensors. We perform the fusion model from two approaches that interpret the scene environment(fig. 1). The first approach considered is the object detector YoloV3[14]. YoloV3 uses RGB images to obtain possible object locations. We used pre-trained weights for the YoloV3 detector. Our second approach obtains the dynamics information in the scene. This Bayesian filter is known as the Conditional Monte Carlo Dense Occupancy Tracker(*CMCDOT*) framework [15]. Both approaches work in real-time and run in parallel. Our main contributions in this work can be summarized as follows:

- Fusion of an object detection method with a Bayesian filter framework at a later stage for recognize moving objects in the environment.

- Since we use a Bayesian filter for detecting moving objects, we do not need additional labeling of dynamic objects for training.

The paper structure is as follows: Section II reviews the related work. Section III details the proposed model for the fusion between one object detector approach with dynamic spatial occupancy estimations. Section IV shows the experimental results and discussion. Finally, section V provides concluding remarks.

II. RELATED WORK

We developed this section in three main categories: *i)* camera sensors, *ii)* dense point clouds from lidar sensors, and *iii)* sensors fusion.

A. Camera sensors

We consider essential aspects of object detections as well as some applications. For example, several deep-learning approaches exist to detect and classify objects in *RGB* images [16], [17], [18], [19], [20], [21], [22], [23], [24], [14]. In this study, we prefer to use YoloV3 [14] due to its real-time performance, as reported by [6]; although, its accuracy may not always achieve the highest performance.

B. Dense point clouds from lidar sensors

We highlight some studies that were looking for moving objects in the scene, only using lidar sensors. For example, Rummelhard et al. [25] developed an ADAS system architecture considering environment representations. The Conditional Monte Carlo Dense Occupancy Tracker(*CMCDOT*) framework is the base of this system. This framework infers dynamics in the scene through a hybrid representation of the environment. These representations consist of static and dynamic occupancy, empty spaces, and unknown areas.

In [11], Wirges develops a deep convolutional object detector for automated driving applications. This model also estimates classification, pose and shape uncertainty of each detected object using a lidar. Chen et al. in [9] developed a new algorithm that runs in real-time to identify and track vehicles even under occluded situations. This algorithm is only to detect vehicles. In [10], Postica et al. presented the results to improve the speed and accuracy of the Dempster-Shafer theory. This theory identifies which point is dynamic or static in an occupancy map generated from a data point cloud.

Dewan, Oliveira, and Burgard in [26] proposed a method for pointwise semantic classification of 3D lidar data into three classes: non-movable, movable, and dynamic. Authors combine information retrieved from a neural network (FastNet) and motion cues from a Bayes filter framework to estimate the pointwise semantic classification. This approach does not recognize the kind of objects in the scene.

C. Sensors fusion

Finally, we describe some studies that consider the use of sensors fusion models in their proposals. For example, Ranft and Stiller in [2] think that RGB cameras have a critical role in vehicular perception. However, they conclude the need to develop a diverse perception system using information from a set of different sensors. In [27], the author presents a literature review on environment perception for intelligent vehicles. Zhu et al. analyze some algorithms for vehicular tracking. They conclude that some of the main problems in this kind of algorithm are the measurement error uncertainty, data association, and necessity to efficiently fuse data from multiple sensors.

Gies, Danzer, and Dietmayer in [28], proposed an environment perception framework. This framework extracts objects from a dynamic occupancy grid map and fuses them with tracks of a labeled Multi-Bernoulli filter. This work uses three sensors: camera, lidar, and radar. The environment perception framework does not run in real-time. In [13], the author proposed an asynchronous approach to fuse contradictory information over time. The fusion model considers the projection of segmented results obtained from a monocular camera with a ground plane derived from lidar data. The proposal did not achieve real-time performance.

Rangesh and Trivedi in [29] proposed a framework to perform full-surround Multi-Object-Tracking. This framework uses a calibrated camera arrays, with varying degrees of overlapping FoVs, and the option to include low-resolution range sensors for accurate localization of objects in 3D. One of the main results obtained in this work is the efficient and fast early-fusion model adopted. The early-fusion model can handle the objects proposed from different sensors inner the calibrated camera array. The proposed framework is also made highly modular. This framework can work with any camera configuration with varying FoVs, and with or without lidar sensors.

Based on the three categories analyzed, the development of a fusion model is the best alternative to recognize moving objects in the scene. In this paper, we propose a fusion model that combines one real-time object detector with a highly-parallelized generic spatial occupancy tracker that requires no training. In the next section, we provided a depth explanation of the fusion model developed.

III. METHOD

We took into consideration for the fusion model the following steps: *i)* selection of a deep-learning approach, *ii)* development of the projective transformations and *iii)* outcomes association.

A. Selection of a deep-learning approach

In the last years, the *deep-learning* has demonstrated significant results in some computer vision tasks, such as object detection, segmentation, among others [30]. Consequently, this machine-learning technique was applied to detect and classify the objects founded in *RGB* images.

Figure 2 shows the *YoloV3* test using our autonomous platform. The *YoloV3* approach implementation is an adaptation from the *ROS package* developed by Bjelonic in [31].

Fig. 2: Object detection and classification using the *YoloV3* approach in our autonomous vehicle.

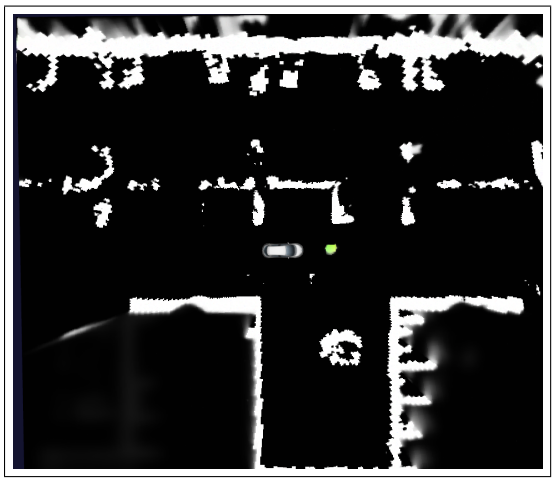

Fig. 3: Occupancy grid map. The white elements in the image represent the static-objects, while the green color represent the only dynamic-object in scene.

B. Development of the projective transformations

In this proposal, the fusion process considers the relation between *geometric* and *semantic* information. The geometric information represents the occupancy grid plane obtained from a lidar input. On the other hand, the semantic information symbolizes some attributes of an object in the image plane obtained from an RGB input. Due to the different planes between the geometric and semantic information, *projective transformations* are necessary to realize the fusion.

We applied the projective transformations from the *occupancy grid plane* to the *image plane*. As aforementioned, there is not a one-to-one correspondence between the grid cells and point cloud; therefore, it not possible to transform the occupancy grid into the image plane via direct registration. We divide the projective transformations into two steps: *i)* the occupancy grid frame transformation to camera frame, and *ii)* camera frame transformation to image plane.

1) Occupancy grid plane transformation to camera plane: As noted earlier, the geometric information represents the occupancy grid plane. By definition, the occupancy grid plane is a two-dimensional spatial map whose grids represent a probabilistic estimation of the occupancy [32]. This spatial map (Fig. 3) is like a two-dimensional matrix where, in each spatial point (x,y), it is possible to obtain one occupancy state. These states are static-object, dynamic-object, empty, and undefined. In our fusion model, we look explicitly for spatial points (x,y) with dynamic-object states. Therefore, geometric information preprocess was necessary to obtain only these dynamic points. This preprocess uses a CUDA implementation.

The dynamic points p_{Di} are the input for the projective transformation ${}^{o}_{c}tf$ between the occupancy grid plane *o* and the camera frame *c*. The main idea with ${}^{o}_{c}tf$ transformation is to find the correspondence from one dynamical point p^{o}_{Di} (i.e.,$p^{o}_{Di}(x^{o}_{i},y^{o}_{i},z^{o}_{i})$) on the occupancy grid plane, inside the camera frame *c*. Therefore, this correlation between the planes *o* and *c* for p^{o}_{Di} is described by:

$$\begin{bmatrix} x^{c}_{i} \\ y^{c}_{i} \\ z^{c}_{i} \\ 1 \end{bmatrix} = {}^{o}_{c}tf \begin{bmatrix} x^{o}_{i} \\ y^{o}_{i} \\ z^{o}_{i} \\ 1 \end{bmatrix} \quad (1)$$

where x^{c}_{i},y^{c}_{i},z^{c}_{i} are the transformed coordinates of the new dynamical points p^{c}_{Di} in the camera frame *c*.

2) Camera frame transformation to image plane: The new projective transformation projects the dynamical points p^{c}_{Di} from the camera frame *c* on the image plane *m*. Equation 2 defines this projection as:

$$\begin{bmatrix} px^{m}_{i} \\ py^{m}_{i} \\ 1 \end{bmatrix} = K \begin{bmatrix} x^{c}_{i}/z^{c}_{i} \\ y^{c}_{i}/z^{c}_{i} \\ 1 \end{bmatrix} \quad (2)$$

where *K* is the camera calibration matrix. px^{m}_{i} and py^{m}_{i} are the pixel coordinates of the projection dynamical points p^{c}_{Di} (i.e., x^{c}_{i},y^{c}_{i},z^{c}_{i}) on the image plane *m*.

C. Outcomes association

YoloV3 approach detects the object present into an image using a bounding box with an associated class score. This bounding box is defined by two points $B^{m}_{min}\,(x^{m}_{min}, y^{m}_{min})$ and $B^{m}_{max}\,(x^{m}_{max}, y^{m}_{max})$ on the image plane *m*. These two points and the result of the projective transformations are now on the same plane. Consequently, the outcomes association is possible. Figure 4 shows the outcomes association obtained from the geometric and semantic information.

Note that in this case the dynamical points p^{m}_{Di} on the image plane *m* (Fig. 4) are in the bottom from bounding box. This behavior is due to the lack of depth data. Some of the dynamic points $p^{m}_{Di}s$ will be below the bottom of the bounding box due to imperfections in the projection since we do not have the exact depth information. Equation 3 define the heuristics considered to look for the $p^{m}_{Di}s$ close to the bounding box bottom limit.

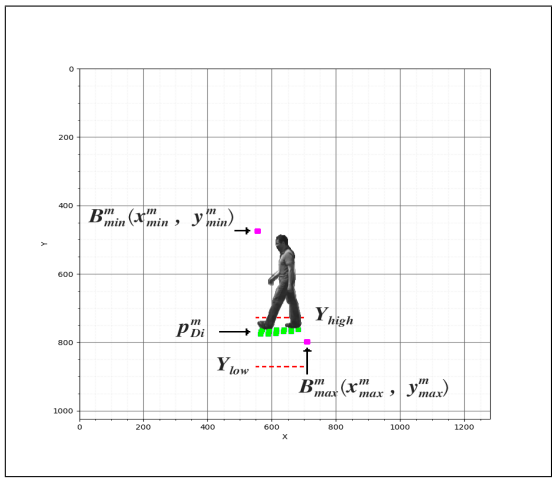

Fig. 4: Outcome association. The purple points in the image represent the two bounding box points, while the green points are the dynamical points projected over the image plane.

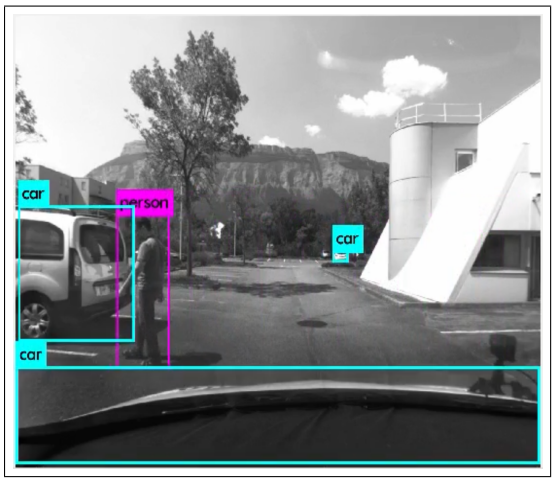

Fig. 5: Overlap case. Figure shows how the purple bounding box(Class person) overlap the blue bounding box (Class car).

$$f_m\left(B^m_{min,max}, p^m_{Di}\right) = \begin{cases} 1, x^m_{min} < px^m_i < x^m_{max} \\ \quad and \quad Y_{high} < py^m_i < Y_{low}. \\ 0, other \quad case. \end{cases} \tag{3}$$

where,

$$\begin{aligned} Y_{high} &= \frac{((8 * y^m_{max}) + y^m_{min})}{9} \\ Y_{low} &= (2 * y_{max}) - Y_{high} \end{aligned}$$

The fusion model function depends on bounding box points $B^m_{min,max}$ and the dynamic points p^m_{Di} by $f_p\left(B^m_{min,max}, p^m_{Di}\right)$. The coordinates x^m_{min}, y^m_{min} and x^m_{max}, y^m_{max} represents the bounding box points B^m_{min} and B^m_{max}(Fig. 4). Moreover, px^m_i, py^m_i indicates the position of each dynamic point p^m_{Di}. Finally, in figure 4 is possible to observe that the dynamical points $p^m_{Di}s$ are generally located on the bottom edge from the bounding boxes. Therefore, the thresholds Y_{high} and Y_{low} are computed to define a specific region, where the fusion model will be looking for the dynamic points.

When the fusion model detects at least one $p^m_{Di}s$ using equation 3, the class information obtained from the bounding box identifies the dynamic object. Therefore, we can recognize the moving object in the scene. However, we will need to consider the overlap cases (fig. 5) of the bounding boxes, because we could find erroneous estimations in the results.

A solution to this fusion mistake is to compare the $B^m_{min,max}$ points from the bounding boxes to determine the overlap condition. Consequently, all the $B^m_{min,max}s$ are compared among them by pairs to define the correct bounding box for the dynamic point. Algorithm 1 shows the heuristics defined to find the overlap condition between two bounding boxes.

Algorithm 1: Overlap case

Data: Bounding boxes from the fusion model outcomes(*bbx*).

$bbx_1 = B^{\ m}_{min1,max1}$ and $bbx_2 = B^{\ m}_{min2,max2}$

$bbx_1 = \langle B^m_{min1}\left(x^m_{min1}, y^m_{min1}\right), B^m_{max1}\left(x^m_{max1}, y^m_{max1}\right)\rangle$

$bbx_2 = \langle B^m_{min2}\left(x^m_{min2}, y^m_{min2}\right), B^m_{max2}\left(x^m_{max2}, y^m_{max2}\right)\rangle$

if $(y^{\ m}_{min1} > y^m_{max2})$ *and* $(y^m_{min2} < y^m_{max1})$ **then**
  return ; /* one bbx to the above */
else if $(y^{\ m}_{min1} < y^m_{max2})$ *and* $(y^m_{min2} > y^m_{max1})$ **then**
  return ; /* one bbx to the below */
else if $(x^{\ m}_{min1} < x^m_{max2})$ *and* $(x^m_{min2} > x^m_{max1})$ **then**
  return ; /* one bbx to the right */
else if $(x^{\ m}_{min1} > x^m_{max2})$ *and* $(x^m_{min2} < x^m_{max1})$ **then**
  return ; /* one bbx to the left */
else
  $\max\langle y^m_{max1}, y^m_{max2}\rangle$; /* bbx overlapped */
end

IV. EXPERIMENTS

We tested our approach using the *KITTI* dataset [33]. The KITTI dataset helped us to validate the YoloV3 detector and its fusion with CMCDOT. We use a KITTI-ROS bag file with the road data category number fifteen in the validation experiments. The execution of this file does not have any temporal restriction. We use the tracklets message from the KITTI-ROS bag to obtain the ground-truth information about the other vehicles in the scene. Our ground-truth considers details from other vehicles as the timestamped, vehicle Id, position (x,y), and classification (e.g., Car, Van, Cyclist, pedestrian).

We use the ground-truth information to evaluate the results of the objects detected for our approach in two steps. The first step evaluates the results obtained only by the YoloV3 detector. The second step assesses the results collected from

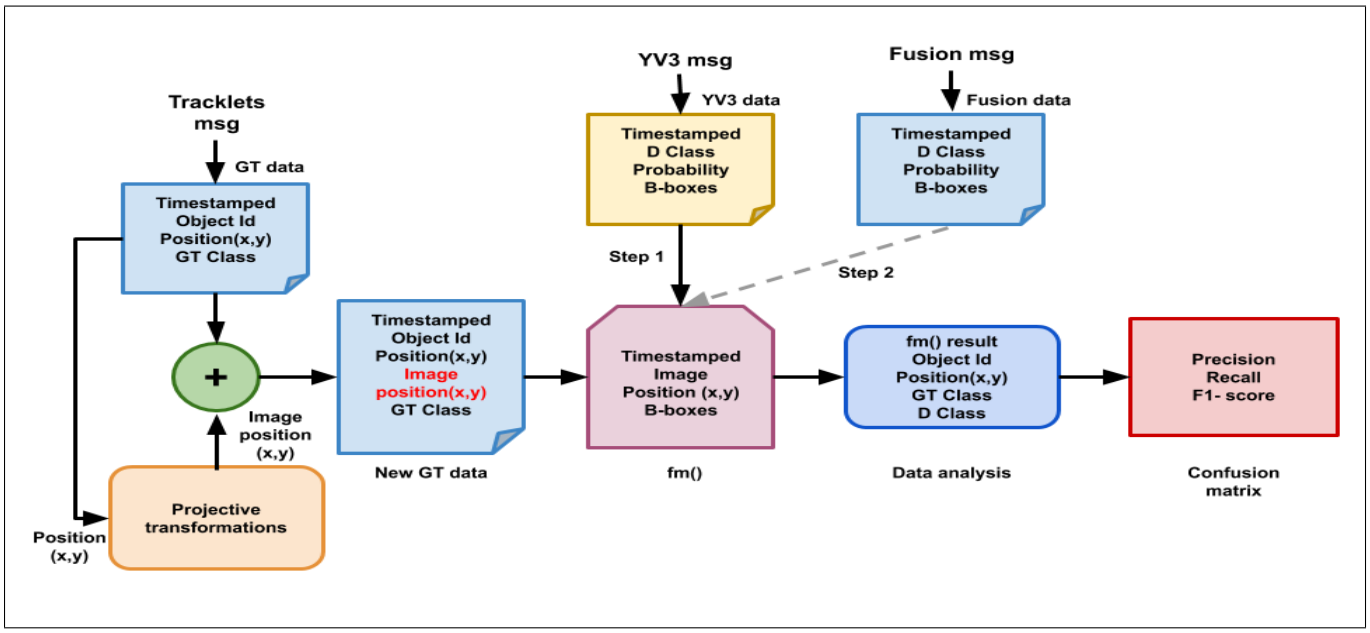

Fig. 6: Validation methodology. Figure presents the block diagram of the methodology applied in our approach. ROS messages (Tracklets, YV3, and Fusion) provide the needed information. That information is compared using the function *fm()*. Data analysis techniques preprocess the results from *fm()*, which are validated using three metrics obtained from a confusion matrix.

the fusion between the YoloV3 detector and the CMCDOT framework. We used the concept of projective transformation applied in III-B to identify each car in the scene, based on the position obtained in the ground-truth information. The evaluation metrics used for the detection in each step are precision, recall, and F1-score by each class detected. Figure 6 shows in detail the validation methodology used for the fusion model.

A. Experimental results

1) YoloV3 detector validation: As can be observed in III-C, YoloV3 detection has a direct consequence on the performance of our approach. Therefore, we execute an experiment to validate the YoloV3 detection based on a raw data sequence from KITTI. The experiment's objective is to know the performance of the YoloV3 detector. Therefore, the validation methodology presents in fig. 6 was defined to achieve our objective. We apply this validation methodology in the YoloV3 detector and the fusion between YoloV3 and CMCDOT.

The base of our methodology is the projective transformations applied in III-B. This mathematical model ubicates the position of a vehicle around the ego-car on the image plane. Therefore, this fact lets us compare the vehicle position with the bounding box obtained from YoloV3 in the image plane. These comparisons can identify a vehicle in front of the ego-car in a specific timestamped during the experiment. Finally, since our approach does not only use an RGB camera to define motion estimation, this methodology lets us validate our approach. The main reason is that the ground truth does not depend on semantic information.

Table I shows the results of the validation for the YoloV3 detector. We used the confusion matrix measure to obtain the performance, recall, and F1-score for each class. The result considers all the objects detected and not detected for YoloV3. The Car class is the best result obtained in our validation for all the metrics. This result is coherent with the KITTI dataset because it is the most frequently observed class in this dataset. Furthermore, the Car-result obtained is practically similar to the result submitted by Jizhi Zhang in the KITTI benchmark[1].

TABLE I: Result of the YoloV3-detector validation.

Class	Precision(%)	Recall(%)	F1-score(%)
Car	100	81.33	89.70
Van	92.85	81.25	86.66
Truck	79.41	90	84.37
Cyclist	100	30.21	46.40

Applying data analysis(fig. 6), we look for the best performance of the YoloV3 detector in function from the position of the other vehicles around ego-car. With this intention, we consider longitudinal distances of twenty to fifty meters with a window size of ten meters, and lateral distances between ten and twenty meters. The best performance found for the YoloV3 detector was thirty meters for longitudinal distance and ten meters for lateral distance. Table II shows the results obtained with this setup. The longitudinal distance founded is practically the same presented for Rangesh and Trivedi in [29]. Note that the class Cyclist is not present in this setup

[1] http://www.cvlibs.net/datasets/kitti/eval_object_detail.php?\&result=326e3ea44e5b02885a14389b26b79d4d85d9033c

since it was not observable in the given longitudinal and lateral distances.

TABLE II: Result of the YoloV3-detector validation with a longitudinal distance equals thirty meters, and a lateral distance equals ten meters.

Class	Precision(%)	Recall(%)	F1-score(%)
Car	100	92.81	96.27
Van	100	83.33	90.90
Truck	100	90.90	95.23

2) Fusion YoloV3-CMCDOT validation: Based on the setup and the results obtained with the validation of the YoloV3 detector, we evaluate the fusion YoloV3-CMCDOT. We again use the validation methodology applied in IV-A.1. Finally, we considered the distance range obtained in the validation of the YoloV3 detector to tuning the dynamic points of the CMCDOT framework in that range. Table III shows the result obtained from the validation of the fusion YoloV3-CMCDOT for three classes.

TABLE III: Result of the Fusion YoloV3-CMCDOT validation using the bounding boxes overlap algorithm.

Class	Precision(%)	Recall(%)	F1-score(%)
Car	100	68.62	81.39
Van	100	33.33	49.99
Truck	100	81.81	89.99

Note that the Recall metric in Table III shows a low value with respect to the Recall result obtained in Table II. The differences among the results(i.e., Car = 24.19, Van = 50, and Truck = 9.09) are explained based on the overlap case. Such as was explained in section III-C, the overlapping bounding boxes produce a fusion mistake. To avoid this mistake, Algorithm 1 chooses from the fusion model outcome(i.e., bounding box) with the y^m_{max} higher. This means that the fusion model does not consider all bounding boxes detected. Therefore, the Recall metric is affected by the overlap case.

Table IV shows the result obtained from the fusion YoloV3-CMCDOT without using the overlap case algorithm. Note that the Recall metric is higher in the Car and Van classes, but lower in the Truck class than the results in Table III. The lower result obtained for the truck class allowed us to conclude that the size of the bounding boxes favors this class over the overlap condition. The result in Table IV demonstrates the direct effect that the bounding boxes overlap algorithm has over the fusion model validation. However, this algorithm avoids the fusion mistake between static and dynamic objects detected when the overlap condition is happening.

TABLE IV: Result of the Fusion YoloV3-CMCDOT validation without use the bounding boxes overlap algorithm.

Class	Precision(%)	Recall(%)	F1-score(%)
Car	100	73.20	84.52
Van	100	75.0	85.71
Truck	100	72.72	84.20

3) Zoe platform validation: Finally, we use our autonomous car platform to test the proposal. This autonomous platform is a Renault Zoe car equipped with a Velodyne HDL64, 3 Ibeo Lux Lidars, GPS-IMU Xsens, and 2 IDS cameras. The Zoe PC has a processor Intel Xeon 2.6GHz with 24 cores, 32 GB of memory, and one Nvidia Geforce GTX Titan X graph card. The YoloV3 detector was training with the MSCOCO dataset for six specific classes(i.e., person, bicycle, car, motorbike, bus, and truck). Figure 7 shows a traffic scene example of the results obtained with our proposal using the Zoe platform in the streets of Montbonnot-Saint-Martin.

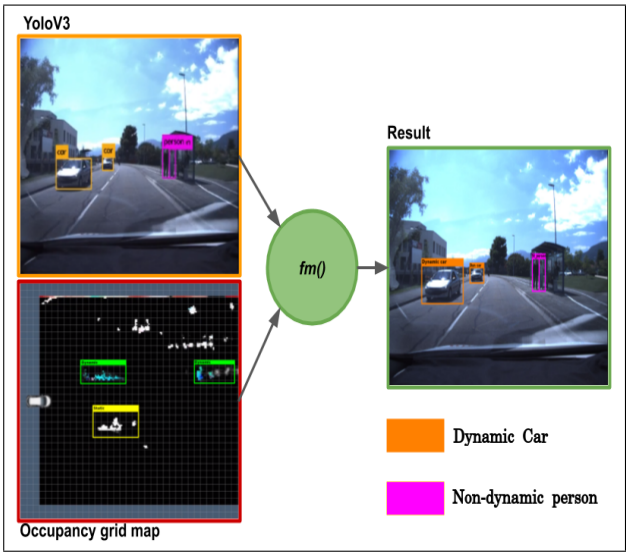

Fig. 7: Zoe platform validation. Figure shows an example of the recognize of two cars moving (orange bounding boxes inside the result block) in a real scene.

Regarding that traffic scene in figure 7 is possible to observe the input and the result of our proposal. Inside the occupancy grid map (red box) is possible to observe three colorful rectangles. The green rectangles represent the dynamic objects in the scene, and the yellow one, the static objects. On the other hand, the YoloV3 detector(orange box) to find two vehicles and two persons in the scene. Finally, The fusion model result (green box) shows the two cars'(orange bounding boxes) recognition in the scene as dynamic objects.

V. CONCLUSION

In this paper, we consider the problem of recognizing moving objects around an autonomous vehicle. We have proposed a fusion model that considers the relation between the occupancy grids and semantic information obtained from lidar, and monocular RGB cameras. This fusion can produce some mistakes when the YoloV3 detector overlaps some object detections in the scene. Consequently, we develop the bounding boxes overlap case algorithm that chooses the bounding box closer to the Ego-car considering the y^m_{max} value.

We divided our validation methodology into two parts. The first part validates the YoloV3 detector to know its performance. The second part evaluates the fusion between the YoloV3 and the CMCDOT framework with/without the bounding boxes overlap case algorithm. Our validation methodology considers the same conditions for the experiments in each part. The performance computes in our validation method favors the detection of one object based on its position in a period of time-specific, using projective transformations. We do not take into consideration determined semantic labels for the validation.

Our future work aims to eliminate the overlap cases from this proposal, given the results obtained in subsection IV-A.2. Therefore, in a new version of the fusion model, we need to consider the static objects in the scene. Furthermore, The new fusion model has to provide the position, velocity, and direction from whatever object around an autonomous vehicle.

ACKNOWLEDGMENT

This work has been supported by the French Government in the scope of the FUI STAR project. We also would like to thank Manuel Diaz Zapata, Thomas Genevois, Anshul Paigwar, Jerome Lussereau, Jean-Alix David and David Sierra Gonzales for their meaningful technical discussions and suggestions.

REFERENCES

[1] J. Guo, U. Kurup, and M. Shah, "Is it safe to drive? an overview of factors, metrics, and datasets for driveability assessment in autonomous driving," *IEEE Transactions on Intelligent Transportation Systems*, 2019.

[2] B. Ranft and C. Stiller, "The role of machine vision for intelligent vehicles," *IEEE Transactions on Intelligent vehicles*, vol. 1, no. 1, pp. 8–19, 2016.

[3] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda, "A survey of autonomous driving: Common practices and emerging technologies," *IEEE Access*, vol. 8, pp. 58 443–58 469, 2020.

[4] M. Siam, H. Mahgoub, M. Zahran, S. Yogamani, M. Jagersand, and A. El-Sallab, "Modnet: Motion and appearance based moving object detection network for autonomous driving," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018, pp. 2859–2864.

[5] Z.-Q. Zhao, P. Zheng, S.-t. Xu, and X. Wu, "Object detection with deep learning: A review," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 11, pp. 3212–3232, 2019.

[6] Z. Chen, R. Khemmar, B. Decoux, A. Atahouet, and J.-Y. Ertaud, "Real time object detection, tracking, and distance and motion estimation based on deep learning: Application to smart mobility," in *2019 Eighth International Conference on Emerging Security Technologies (EST)*. IEEE, 2019, pp. 1–6.

[7] T. Nowak, M. R. Nowicki, K. Ćwian, and P. Skrzypczyński, "How to improve object detection in a driver assistance system applying explainable deep learning," in *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2019, pp. 226–231.

[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, 2019.

[9] T. Chen, R. Wang, B. Dai, D. Liu, and J. Song, "Likelihood-field-model-based dynamic vehicle detection and tracking for self-driving," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 11, pp. 3142–3158, 2016.

[10] G. Postica, A. Romanoni, and M. Matteucci, "Robust moving objects detection in lidar data exploiting visual cues," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 1093–1098.

[11] S. Wirges, M. Reith-Braun, M. Lauer, and C. Stiller, "Capturing object detection uncertainty in multi-layer grid maps," in *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2019, pp. 1520–1526.

[12] J. Del Ser, E. Osaba, J. J. Sanchez-Medina, I. Fister, and I. Fister, "Bioinspired computational intelligence and transportation systems: A long road ahead," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 2, pp. 466–495, 2020.

[13] E. Capellier, F. Davoine, V. Fremont, J. Ibanez-Guzman, and Y. Li, "Evidential grid mapping, from asynchronous lidar scans and rgb images, for autonomous driving," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, 2018, pp. 2595–2602.

[14] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[15] L. Rummelhard, A. Nègre, and C. Laugier, "Conditional monte carlo dense occupancy tracker," in *2015 IEEE 18th International Conference on Intelligent Transportation Systems*. IEEE, 2015, pp. 2485–2490.

[16] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[17] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.

[18] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.

[19] J. Dai, Y. Li, K. He, and J. Sun, "R-fcn: Object detection via region-based fully convolutional networks," in *Advances in neural information processing systems*, 2016, pp. 379–387.

[20] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[21] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*. Springer, 2016, pp. 21–37.

[22] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7263–7271.

[23] Z. Li and F. Zhou, "Fssd: feature fusion single shot multibox detector," *arXiv preprint arXiv:1712.00960*, 2017.

[24] S. Liu, D. Huang, *et al.*, "Receptive field block net for accurate and fast object detection," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 385–400.

[25] L. Rummelhard, J. Lussereau, J.-A. David, C. Laugier, S. Dominguez, G. Garcia, and P. Martinet, "Perception and automation for intelligent mobility in dynamic environments," 2017.

[26] A. Dewan, G. L. Oliveira, and W. Burgard, "Deep semantic classification for 3d lidar data," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 3544–3549.

[27] H. Zhu, K. Yuen, L. Mihaylova, and H. Leung, "Overview of environment perception for intelligent vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 10, pp. 2584–2601, 2017.

[28] F. Gies, A. Danzer, and K. Dietmayer, "Environment perception framework fusing multi-object tracking, dynamic occupancy grid maps and digital maps," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, 2018, pp. 3859–3865.

[29] A. Rangesh and M. M. Trivedi, "No blind spots: Full-surround multi-object tracking for autonomous vehicles using cameras and lidars," *IEEE Transactions on Intelligent Vehicles*, vol. 4, no. 4, pp. 588–599, 2019.

[30] MathWorks, "*What Is Deep Learning? 3 things you need to know*," 2019, https:\/\/www.mathworks.com\/discovery\/deep-learning.html [Accessed: 30082019].

[31] M. Bjelonic, "YOLO ROS: Real-time object detection for ROS," https://bit.ly/2v1jSwi, 2016–2018.

[32] Ö. Erkent, C. Wolf, C. Laugier, D. S. González, and V. R. Cano, "Semantic grid estimation with a hybrid bayesian and deep neural network approach," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 888–895.

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