Fusion of Data from Lidar and Camera in Self Driving Cars

Mohamed Ahmed
Robotics Institute
Innopolis University
Innopolis, Russia
o.ahmed@innopolis.university

Alexandr Klimchik
Robotics Institute
Innopolis University
Innopolis, Russia
A.Klimchik@innopolis.ru

Riby Abraham Boby
Mechanical Engineering
IIT Madras
Chennai, India
ribyab@gmail.com

***Abstract*—Sensor data fusion is one of the important solutions for the perception problem in self-driving cars, the main aim is to enhance the perception of our system without losing real-time performance and therefore,it is a trade-off problem and its often observed that most models that have a high environment perception cannot perform in a real-time manner.**

In this paper we discuss how we can address this problem using a 3D detector model (Complex-Yolov3) and a 2D detector model (Yolo-v3) , then applying the Image-Based Fusion method that could make a sensor fusion between Lidar & camera information with a fast and efficient late fusion technique that is discussed in detail in this paper.

Then we use the mean average precision metric in order to evaluate our object detection model and to compare the proposed approach with them as well.

In the end, we show the results on the Kitti data set as well as our real hardware setup, which prove that our proposed approach could work efficiently in a real-time manner.

I. Introduction

Object detection is a fundamental problem in many fields and has a huge impact on self-driving cars, relevant reliability and safety can be achieved with the help of different sensors such as ultrasonic, cameras, radars and Lidars mounted on vehicles with redundancy resolution techniques and sensor fusion algorithms.

Classification approaches used in recent years have focused on image recognition research. To produce proposals for bounding boxes, these approaches generate object proposals such as sliding windows [1], edge boxes [2], choose search [3], Multiscale Combinatorial Grouping (MCG) [4], and then utilize a CNN pipeline [5], [6] to perform recognition for the suggested object region. The high computational cost is a typical drawback. Furthermore, cameras lack information on the 3D location, orientation, and shape of objects, as well as fluctuating lighting levels, which results in inaccurate object region proposals.

Using the complementary information offered by LIDAR and cameras to obtain very precise object positions and classifications for self-driving cars is one solution. In other words, good fusion techniques can play a great role in minimizing the disadvantages of both sensors and allows autonomous vehicles to work in real-time with accurate precision for object detection.

II. Literature Review

Many recent researches have focused on the merging of data from multiple sensors. A typical method is to merge the LIDAR point cloud data with the camera images at the pixel level, with a matching RGB color pixel for each LIDAR point within the image [7]. Another approach is to take the data features of each sensor and combine them to identify and track moving objects [8] -[9]. They introduced a Multi-View 3D network (MV3D) for 3D object recognition in [10], which integrates several views of LIDAR point cloud data with images to propose and classify 3D objects. For small object classes, an enhanced deep learning model called AVOD (Aggregate View Object Detection) [11] has been presented that multimodally fuses data provided by point cloud and images to build high-resolution feature maps for the production of trustworthy 3D object suggestions. They use continuous convolutions to fuse LIDAR and image feature maps at various resolution levels for 3D object recognition in [12]. Rather than recognizing things independently from LIDAR point clouds or images, this method combines the final findings acquired by the two sensors.

Sensor fusion is one of the important solutions for perception problem in self-driving cars and autonomous systems in general for many aspects like localization, perception, etc. We need to improve the perception of our system without losing real-time performance. At the same time, it is a trade-off problem where most of models that have high environmental perception cannot perform in a real-time manner and vice versa for models which can work in real-time will neglect important information which is necessary for enhancing the perception. Therefore, it is essential to continue tracking the performance of the developed sensor fusion technique to not lose either perception or real-time performance at the cost of the other.

III. Methodology

A. Complex-Yolo Model (Lidar)

For the 3D object detector, we use the Complex-YOLOv3 model. It works by prepossessing the Lidar point-cloud data and transforms them to a birds-eye-view (bev) RGB-map. The Complex-YOLO network takes the bev RGB map as input.

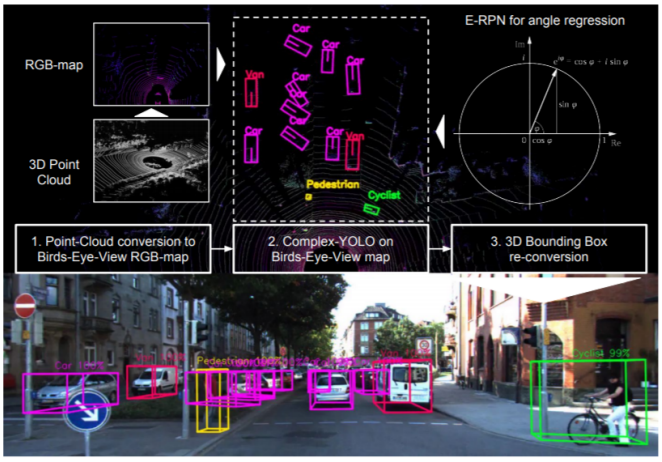

Figure 1. 3D object detector model (Complex-Yolo) for point cloud working idea and results [13]

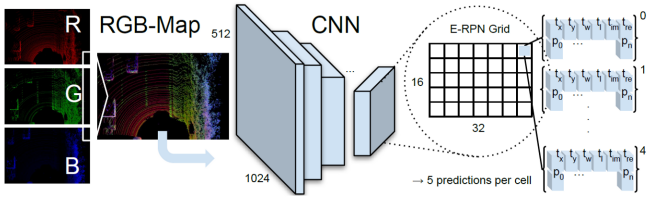

Figure 2. 3D object detector (Complex Yolo) Architecture [13]

It uses a simplified YOLOv3 CNN architecture extended by complex angle regression and E-RPN (Euler Region Proposal Network) to detect accurate multiclass-oriented 3D objects while still operating in real time.

B. Yolo-v3 Model (Images)

For 2D objects detector, we will use the Yolo-v3 model, as it gives less inference time when working with images in the KITTI dataset (15 ms) compared to the Yolo-V4 model with inference time (45 ms), we will also not resize the input because resizing the inputs gives poor results when we tried to do so.

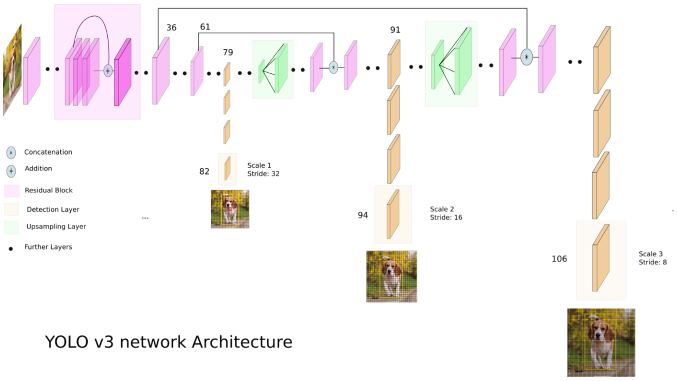

Figure 3. 2D object detector (Yolo-V3) Architecture

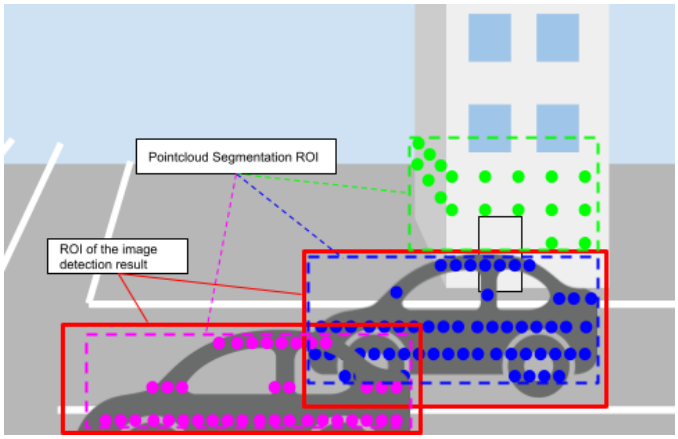

Figure 4. Image Based Fusion

C. Image Based Fusion

1) Summary of the working algorithm: We are using bounding boxes obtained from 3D object detector (Complex-yolo) that are less likely to be objects and overwriting labels of those objects with that Region Of Interests (ROIs) by a 2D object detector (Yolo-v3).

The detected objects from the 3D object detector are then projected onto image planes, and then if the ROIs of clusters and ROIs by a detector are overlapped, the labels of clusters are overwritten with those of ROIs by the 2D object detector. The Intersection Over Union (IoU) is used to determine whether there are overlaps between them.

The advantage in image-based fusion is that we avoided the other problems of the above methods of taking fusion to higher dimension space as the point cloud-based fusion and also we avoided problems of associations of ROI through different frames as the Kalman filter, tackling our problem in a 2D dimension in which concerning the current frame only makes it less computationally expensive to do the fusion and gets higher perception with being able to work in real time.

2) Adding Feature of getting objects distance: We used the point cloud projected onto the image and mapped it to the bounding boxes that are output from the Yolo-V3 model.

The problem is that the lidar data is sparse, so not every pixel in the image will have a correspondent point from lidar. We managed to tackle this problem by using the Nearest-Neighbor technique.

The bounding box of the Yolo-V3 model is an array of 4 values $[x, y, w, h]$. we are interested in the center of the object that we will donate it as C, where $C = (x + \frac{w}{2}, y + \frac{h}{2})$. We will then try to find the assigned lidar projected point for C, but since lidar is sparse, most probably we will not find a point assigned to this center pixel C, so we need to search for the nearest point to C. We will donate the projected point cloud to the image plane as a vector P, where $P = [p_1, p_2 ... p_n]$ now we will search for the nearest element in P, which is $[p_1, p_2 .... p_n]$ vector to point C which is (C_x, C_y) so we will try find minimum distance from C to p_i $distance = \sqrt{(C_x - p_{xi})^2 + (C_y - p_{yi})^2}$

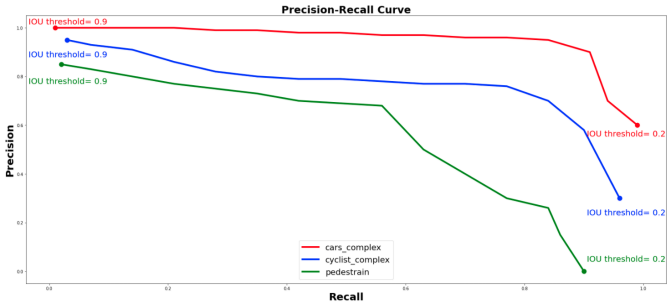

Figure 5. Precision-Recall Curve of 3D object detector (Complex-Yolo) model for classes Cars, Cyclist & Pedestrians

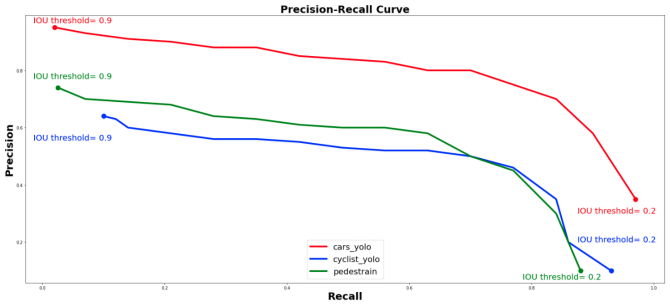

Figure 6. Precision-Recall Curve of 2D object detector (Yolo-v3) model for classes Cars, Cyclist & Pedestrians

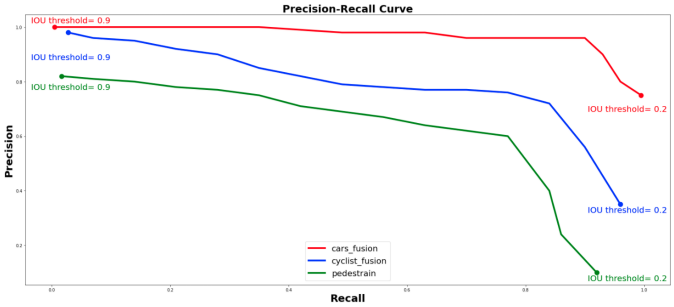

Figure 7. Precision-Recall Curve of the fusion for classes Cars, Cyclist & Pedestrians

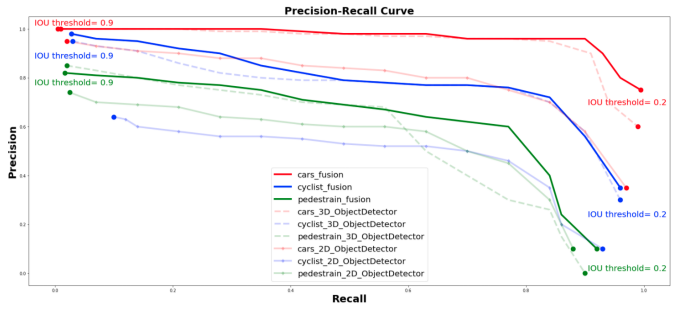

Figure 8. Precision-Recall of Figure 5, 6, 8 in one Figure for comparison

IV. Evaluation and Discussion

For evaluation of our models, we use the average precision metric (AP) and frames per second (FPS). For more information, refer to the appendix.

V. Evaluation of Complex-Yolo

After 220 epoch of training for the tiny Complex-Yolo, which took 2 days,the following results are obtained. In Figure 5 is the precision recall curve from which we obtain the average precision of each class.

In table I are the average precision of each class and then the average of all classes and frames per second for the model.

VI. Evaluation of Yolo-V3

After 2000 epoch of Yolo-V3 training, which took 4 hours, the following results are obtained. In Figure 6 is the precision recall curve from which we obtain the average precision of each class.

In table I are the average precision of each class and then the average of all classes and frames per second for the model.

VII. Evaluation of Image Based Fusion

After applying image-based fusion in we could obtain the following results.

In Figures 8 are the precision recall curves that we calculate the average precision.

In Table I is the average precision of each class, and then the average of all classes and the processing speed of the model.

In Figure 8 the red curve we can see that its the same as in Figure 5 but with higher precision and the recall shift towards right slightly, which is logical, as the average precision of complex-yolo for cars is already so high 0.9625 but after the fusion some true positives are added over the same ground truth bounding box number, and the precision will mostly be the same as the complex-yolo gives high precision already, and that is to prove that the fusion can give better results as the final average precision of the class cars is raised by 1% to 0.9725.

In Figure 8 blue curve we can see that it could address the low precision at some point of the complex yolo model for the cyclist class as shown in 5 with the help of the yolo-v3 model, its standalone performance was not as high according to the precision recall curve 6, which demonstrates that fusion can give better results as the final average precision of the class cyclist increased from 0.7756 of the complex yolo and 0.4196 of yolo-v3, to 0.7985 after fusion, which is better than both models and gives an increase as 2% from the best result of the complex yolo model.

In the green curve in Figure 8 we can see that it takes the same shape as in Figures 6 5 starting from one, but gives a slightly better average precision, the complex-yolo and yolo-v3 have an average precision of 0.5440 and 0.4911 respectively. Fusion could give better results by raising the average precision from 5% to 0.5943.

As we can see, the image and 2D range data fusion give results better than the 2 models although it uses them to fuse information and give better results which show the power of fusion of the data in enhancing the performance of the

Table I
FUSION EVALUATION

Model/Class	Cars	Cyclist	Pedestrians	Average	FPS
Complex-Yolo	0.9626	0.7756	0.5440	0.7607	50
Yolo-v3	0.7847	0.4196	0.4911	0.5651	66
Image Based Fusion	0.9725	0.7985	0.5943	0.7884	28

perception task problem.

Also, the speed dropped to 28 FPS because the models are detecting sequentially, but in future work we will add parallel threads, which will make both models work in parallel, so the FPS will rise to 50 again for the proposed approach.

A. Results Visualization

1) KITTI Dataset: In the KITTI dataset these 7481 frames, the frames are the RGB images taken from the camera and the corresponding Lidar point-cloud data. Frames have different locations and time stamps. We split those frames into training and testing. Figures {9,10,11,12,13,14,15,16} visually show the results of our models.

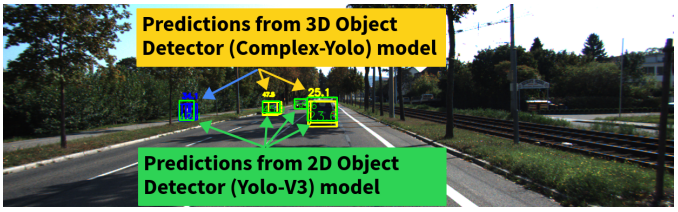

Figure 9. Camera front View

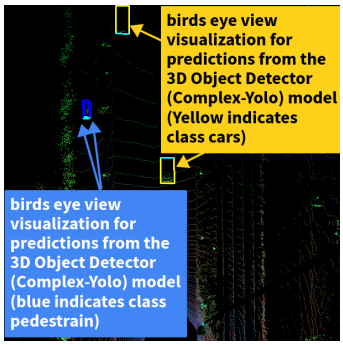

Figure 10. Frame 1 results visualization

In Figure 9 the green squares and the distance written are the predictions of yolo-v3 that work with the image data, the yellow and blue squares and the distance are the predictions of yolo-complex that work with the lidar data. So when one of them fails at some point, the other can support detecting the object.

In Figure 10 is the birds-eye view RGB map of the lidar point cloud, it is the same frame as in Figure 9, the green color indicates height , the blue intensity of the reflected lidar signal, the red is the density of points, yellow, and green squares are predictions from the complex-yolo model.

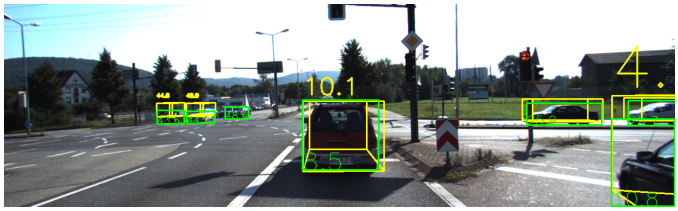

Figure 11. Camera front View

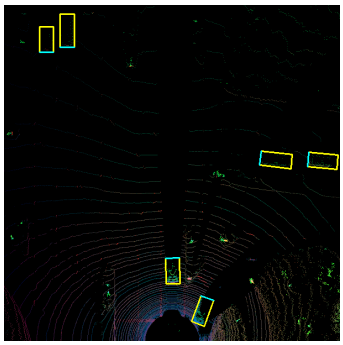

Figure 12. Frame 2 results visualization

In Figure 12 we can see how sometimes the yolo-complex model fails and the yolo-v3 model can support and give even comparable results.

The failure is sometime due to the farther distance of the object such that it will be hardly detected by Lidar as the rays fired from the Lidar diverge and that makes far objects have comparatively lesser points than near objects.

In Figure 13 we can see that the distance difference between the two approaches has an error of 0-2 meters, depending on

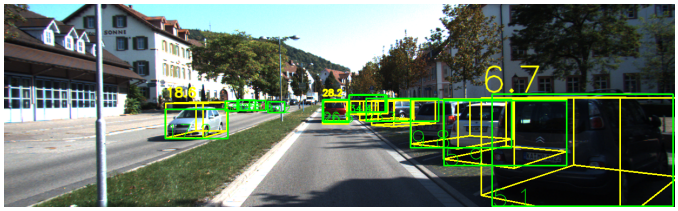

Figure 13. Camera front View

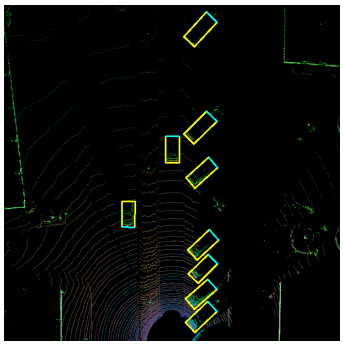

Figure 14. Frame 3 results visualization

Figure 15. Camera front View

Figure 16. Frame 4 results visualization

two factors:

1) The nearest-neighbor point of the 2D approach which gets the distance from the front part of object
2) the 3D approach gets the distance from the center of the object , so in most cases the distance from the 2D approach will have a smaller distance.

In Figure 15 we can see that 3D approach (Complex Yolo) totally fail , which maybe due to different reasons as far distance of object or weather conditions which may affect Lidar sensor reading , only the 2D approach give us results which is why its important to use Image based fusion , so they support and give better results if they both detect same object , if not then we will have higher detection rate so we don't miss an object undetected for a safe environment perception of self driving cars.

2) Real Data: Figures 17 18 19 show a person testing the algorithm on a real hardware lidar and camera, the fusion was performed and the person was detected correctly as a pedestrian.

In Figure 17 we can see that a false object was detected but lidar didn't detect it , and due to low confidence it was neglected by the fusion algorithm.

In Figures 18 and 19 the 2D object detection and the 3D object detection models could detect the person as a pedestrian successfully with the effect that the person was annotated as a pedestrian and the position was displayed later.

VIII. Conclusion

The approaches that work with point cloud directly have higher average precision; however, they lack real-time performance most of the time. On the other hand, approaches that transform point cloud to bev RGB-map formats suffer from information loss, which results in lower average precision but

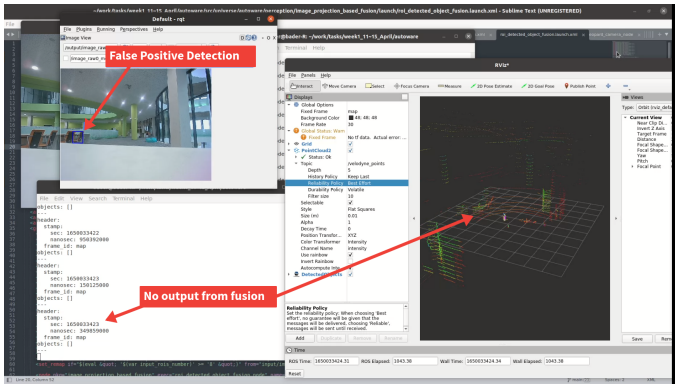

Figure 17. False Detection

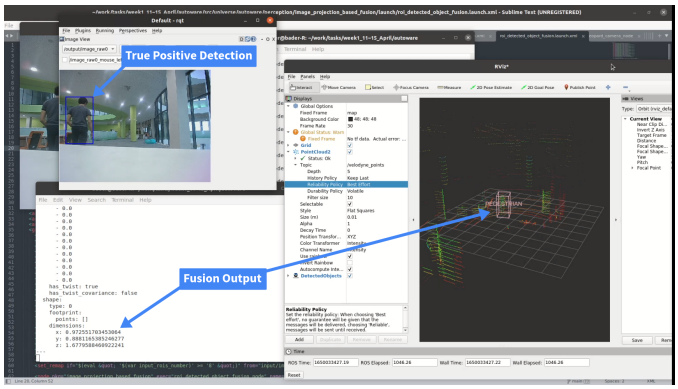

Figure 18. Positive Detection

better real-time performance, and some models as in Complex-Yolo can give a good compromise by giving fair results and still work in real-time. Object detection for 2D problem has been significantly improved through the last decade, and there exist models that can give fair results as well and work in real-time as SSD (e.g yolo-v3). Fusion between previous two approaches is one of the best solutions for the problem in self-driving cars, and there has been significant interest in this area to enhance the autonomous cars and mobile robot systems for better perception of the environment. We passed data from

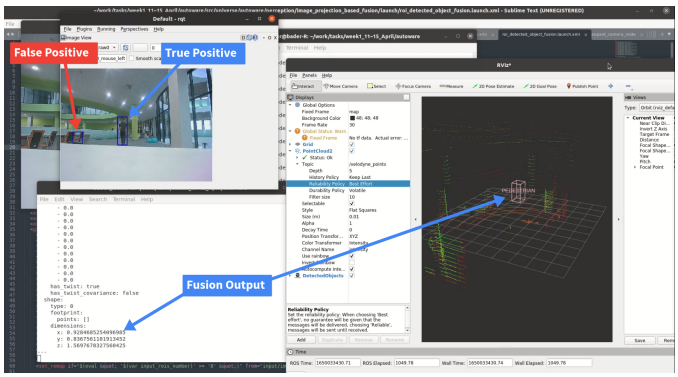

Figure 19. Positive Detection

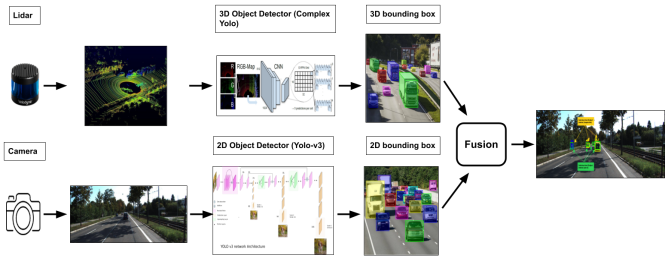

Figure 20. Image Based Fusion

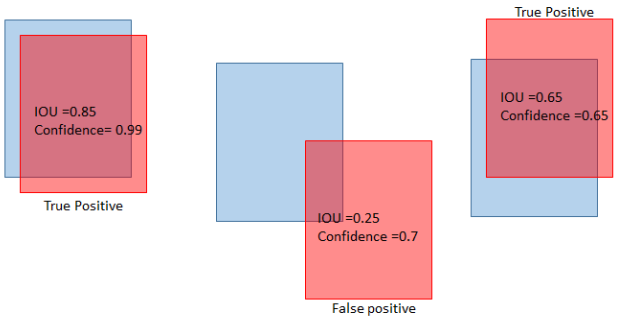

Figure 21. Precision-Recall

Lidar to the 3D object detector model (Complex-Yolo) and evaluated it as a standalone model, also we did the same with Camera and passed the images to the 2D object detector model (YOLO-v3) and evaluated as a standalone model as well.

In our approach (Image-Based Fusion), as we can see in Figure 20, we could see how it combines information to obtain better results than the 2D object detection model and the 3D object detection model, which show the power of data fusion to improve the performance of the perception task problem for self-driving cars.

However, the processing speed decreased to 28 FPS, which can be improved by parallel processing. The proposed methods have been implemented on the KITTI dataset as well as custom generated datasets. The results show that the proposed method enhances the results of individual methods that use point-cloud data or image data.

IX. Appendix

For average precision, it is calculated as follows. After the final predictions are determined, the predicted bounding boxes could be measured against the ground-truth bounding boxes.

In order to calculate the mean average precision (mAP) for each class and see how the object detector is doing; we will first need to calculate the precision and recall for each class.

To do so, the number of true positives must be identified. If a predicted bounding box overlapped a ground truth bounding box by an IOU threshold (0.5), it is considered a successful detection and the predicted bounding box is a true positive. If a predicted bounding box overlapped a ground truth by less than the threshold, it is considered unsuccessful detection, and the predicted bounding box is a false positive. Precision and recall can be calculated from true and false positives, as shown in Figure 21

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} = \frac{count(True\ Positives)}{count(all\ red\ boxes)} = \frac{2}{3}$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} = \frac{count(True\ Positives)}{count(all\ red\ boxes)} = \frac{2}{3}$$

We need to get precision and recall at every IOU threshold and then average it for each class, and then average it again between all classes to get the mAP for the model.

When a model has high recall but low precision, the model classifies most of the positive samples correctly, but it has many false positives (i.e., classifies many negative samples as Positive). When a model has high precision but low recall, then the model is accurate when it classifies a sample as Positive, but it may classify only some of the positive samples.

So, we need to find the threshold that gives us the best of both; the average precision is the area under the curve of the precision-recall curve.

References

[1] P. Dollár, R. Appel, S. Belongie, and P. Perona, "Fast feature pyramids for object detection," *IEEE transactions on pattern analysis and machine intelligence*, vol. 36, no. 8, pp. 1532–1545, 2014.

[2] C. L. Zitnick and P. Dollár, "Edge boxes: Locating object proposals from edges," in *European conference on computer vision*, Springer, 2014, pp. 391–405.

[3] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *International journal of computer vision*, vol. 104, no. 2, pp. 154–171, 2013.

[4] J. Pont-Tuset, P. Arbelaez, J. T. Barron, F. Marques, and J. Malik, "Multiscale combinatorial grouping for image segmentation and object proposal generation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 1, pp. 128–140, 2016.

[5] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Region-based convolutional networks for accurate object detection and segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 1, pp. 142–158, 2015.

[6] X. Chen, K. Kundu, Y. Zhu, H. Ma, S. Fidler, and R. Urtasun, "3d object proposals using stereo imagery for accurate object class detection," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 5, pp. 1259–1272, 2017.

[7] J. R. Schoenberg, A. Nathan, and M. Campbell, "Segmentation of dense range information in complex urban scenes," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2010, pp. 2033–2038. DOI: 10.1109/IROS.2010.5651749.

[8] H. Cho, Y.-W. Seo, B. V. Kumar, and R. R. Rajkumar, "A multi-sensor fusion system for moving object detection and tracking in urban driving environments," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 1836–1843. DOI: 10.1109/ICRA.2014.6907100.

[9] S.-I. Oh and H.-B. Kang, "Object detection and classification by decision-level fusion for intelligent vehicle systems," *Sensors*, vol. 17, no. 1, 2017. [Online]. Available: https://www.mdpi.com/1424-8220/17/1/207.

[10] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3d object detection network for autonomous driving," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul. 2017.

[11] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander, "Joint 3d proposal generation and object detection from view aggregation," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 1–8. DOI: 10.1109/IROS.2018.8594049.

[12] M. Liang, B. Yang, S. Wang, and R. Urtasun, "Deep continuous fusion for multi-sensor 3d object detection," in *Proceedings of the European Conference on Computer Vision (ECCV)*, Sep. 2018.

[13] M. Simon, K. Amende, A. Kraus, *et al.*, "Complexer-yolo: Real-time 3d object detection and tracking on semantic point clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 0–0.