Innopolis University

PROJECT REPORT

Computer Vision Spring 2022 – Master's in RoCV

Supervised by

Prof. Adil Khan

Eng. Karam Almaghout

Submitted by

Ahmed Mohsen Ali a.ali@innopolis.univeristy

Sohaila Ahmed Rabie s.hussein@innopolis.univeristy

19/02/2022

Table of Contents

List of Figures

List of Tables

I. Introduction

Nowadays, introducing autonomous systems have been crucial to every field especially in transportation, where exists the autonomous vehicles and self-driving cars. The presence of autonomous vehicles in the urban transportation system increases the safety of the environment and has potential to reduce accidents [7].

The navigation of an autonomous vehicle consists of three main parts: perception, planning, and control. Due to the availability of large data of pictures and the high computational power that is experienced in recent years, the role of computer vision became essential specifically in the perception task in AVs [2].

In this project, we are interested in executing an accurate path trajectory for an autonomous vehicle. This is considered a challenging task as it requires detection of both static and dynamic objects in the environment in real-time. It also entails executing fast decisions regarding maneuvers, avoiding obstacles, and dealing with emergency situations [5].

Moreover, it requires a precise perception module including a variety of sensors such as: camera, Lidar, …etc., to increase the awareness of the autonomous vehicle and pursue the object detection task in real time with high-level of performance. Since this task involves dealing with specific type of data such as: videos and images, it requires processing and filtering out important information relevant to the task. Hence, as stated before, the computer vision techniques and algorithms play a critical role in solving this problem [2].

The main idea and contribution of the project is to create 3D bounding boxes through Complex-YOLO model and combine it with the 2D bounding boxes generated by the YOLOv3 model. The input data to the Complex-YOLO model is LiDAR point cloud, while the input data to the YOLOv3 is camera images. The combination between the LiDAR and camera sensors will ensure a backup to the object detection task, in case either of the two sensors is faulty. Also, the combined output of the sensors can be compared to ensure obtaining accurate distances to different objects [10],[13].

II. Related Work

There have been several improvements and additions done to the basic YOLO model such as:[1],[4],[9],[10],[11],[13] to enhance the speed of detecting an object and apply real-time object detection. The YOLOv2 has been adjusted to overcome certain drawbacks of YOLO such as: detecting multi-scale objects by using grid cells of size 13x13 instead of 7x7 [4],[9]. Moreover, the third version of YOLO has been improved by depending on logistic regression (classification) rather than conventional regression (SoftMax layer). Hence, the YOLOv3 can perform multi-label classification [10].

However, all the mentioned YOLO models deal only with camera images or videos. Also, it detects objects in a 2D format and creates 2D bounding boxes. On the other hand, Complex-YOLO deals with LiDAR point cloud data and can detect objects in a 3D format and create 3D bounding boxes [13]. In addition, the camera has a limited field of view

(FoV), in which it can detect objects in front of the car only, while the lidar sensor can detect all space around the car.

Regarding 3D point cloud processing, there are multiple algorithms that process and classify objects from such type of data such as Frustum based network and LaserNet. However, these algorithms suffer from low inferencing speed (low fps) or dependence on multiple sensors (Lidar and camera). Therefore, we chose Complex-YOLO to be the main algorithm for our project since it has very high fps rate (50 fps) and depends only on lidar data. It also detects, classify, and add distance information to each object. Such desired features will make it possible for real time implementation for object detection and classification.

Since real time application can not rely on one sensor for data acquisition, we need another approach that can utilize data acquired from two different sensors such as: camera and Lidar. Consequently, the contribution of our project is to use two different models: Complex-YOLO which relies on LiDAR data, and YOLOv3 which depends only on camera data. This combination will account for any sensor malfunction and the combined results will be more accurate than using either model or sensor solely.

III. Idea Implementation

As stated above, we use Complex-YOLO for object detection and classification by utilizing LiDAR data. The main implementation steps for the algorithm are described in the figure below:

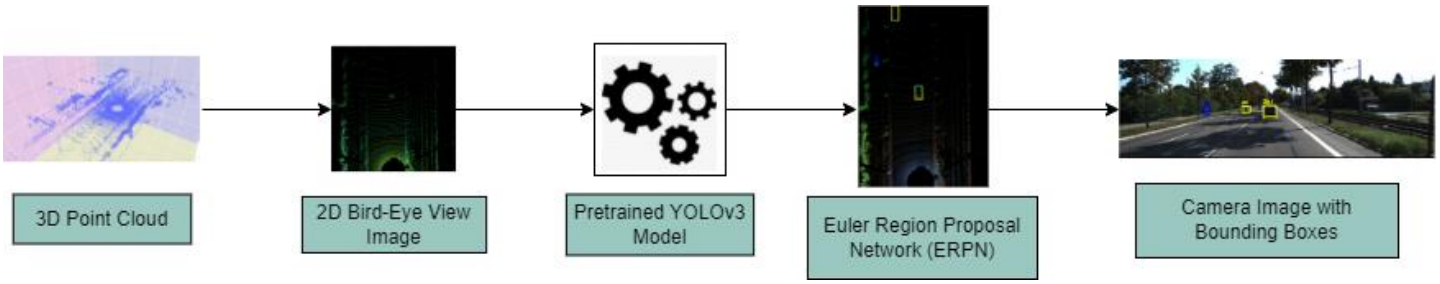

Figure 1: *Results for Complex-YOLOv3 implementation*

Firstly, preprocessing is applied to the raw LiDAR data and then fed into pretrained YOLOv3 model . Secondly, the bounding boxes are reprojected into RGB camera images for visualization. The model steps are explained in detail as follows:

III.I. Dataset Download & Preprocessing

We used a sample of the KITTI dataset since it is widely used in autonomous vehicles applications and contains multiple 3D points clouds along with vehicle images. The data and model used in the project are uploaded in a drive link and their content is explained in the table below:

Folder	Content
Velodyne	contains 3D point cloud
Image	contains RGB images obtained from camera (used for visualization)
Label	contains the ground truth bounding boxes and labels
Calibration	contains transformations between LiDAR coordinates and camera coordinates

Table 1: *Content of the files in the drive link.*

The next step is to convert the 3D point cloud, which contains x, y, z coordinates and reflection rate, into Bird Eye View (BEV) image. The BEV is a regular RGB image, but its channels accounts for height, intensity, and density. The generated BEV is shown below.

Figure 2: *BEV image*

III.II. YOLOv3 Pretrained Model

The Complex-YOLO internally uses the YOLOv3 model for generating preliminary 2D bounding boxes on the BEV image. The output of this step is passed on to ERPN.

The model was pretrained according to the following parameters:

Input Image size	608 x 608 x 3
Batch size	64
Momentum	0.9
Decay	0.005
Learning rate	0.001
Epochs	300

***Table 2**: Pretrained model parameters*

We used the model with these weights, and it provided good results with our dataset.

III.III. Euler Region Proposal Network (ERPN)

One important feature in Complex-YOLO is ERPN as it corrects the orientation of the resulting bounding boxes to project and transform 2D boxes into 3D. The YOLOv3 model output is fed into ERPN, which divide the image into 16 and 32 grid cells. Each grid cell is responsible for 5 bounding boxes. Each bounding box has the following parameters:

- Objectness Probability (whether the bounding box has object or not)
- 3 class probabilities (car, cycle, pedestrian)
- 6 parameters for x, y, width, height, real angle, imaginary angle.
- 5 additional parameters (alpha, cx, cy, pw, pl)

So, each grid cell is responsible for 5 * (6 +1 +3 +5) = 75 features.

III.IV. Results

The resulting bounding boxes are projected into camera images for better visualization. So, for each test image, we get two images with bounding boxes (one is BEV, and the other is from the camera). The results contain the following information:

- Bounding box around each object
- Distance information
- Color of box indicate the class for object (red for pedestrian, yellow for car, blue for cyclist.

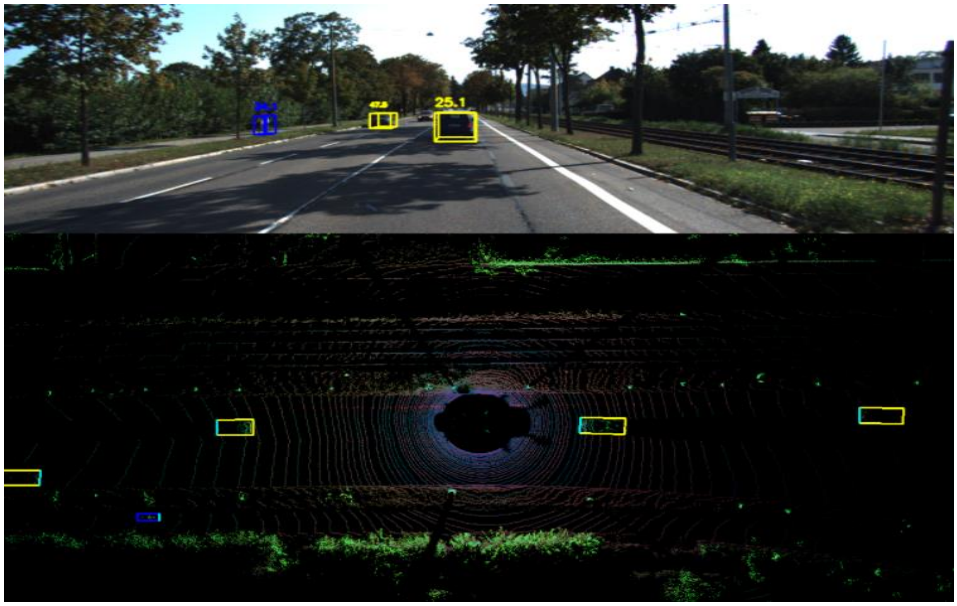

Figure 3: *Model result, camera image (above) and BEV image (below*

Metric Evaluation

We selected the mean average precision (mAP) as our evaluation metric, since it is a widely used assessment method for object classification and detection models i.e., YOLO. The metric evaluation results for Complex-YOLO model are shown below: (IoU Threshold: 0.5)

Class	AP
Car	1.0
Pedestrian	0.89
Cyclist	0.66
Model mAP	0.85

Table 3: *The mean average precision metric evaluation for the Complex-YOLO model.*

Remark:

- The pretrained Complex-YOLO model used in this project can be found in the GitHub repository.
- The link to our code.

Final Expected Outcomes

The final expected outcomes of this project are:

- Implement the pretrained Complex-YOLO for generating 3D bounding boxes and automatically determine the distance to objects. [Finished Step]
- Implement the pretrained YOLOv3 for generating 2D bounding boxes for object detection and classification. [Unfinished Step]
- Combining the YOLOv3 2D bounding boxes with LiDAR 3D point cloud information, to find distances to objects. [Unfinished Step]

Applying two separately distinct approaches to determine the distance of objects is significant, since each approach depend on a different sensor which is helpful in case of sensor malfunction. Besides, the output distance calculated from both models will be compared to ensure accurate determination of the distance of an object.

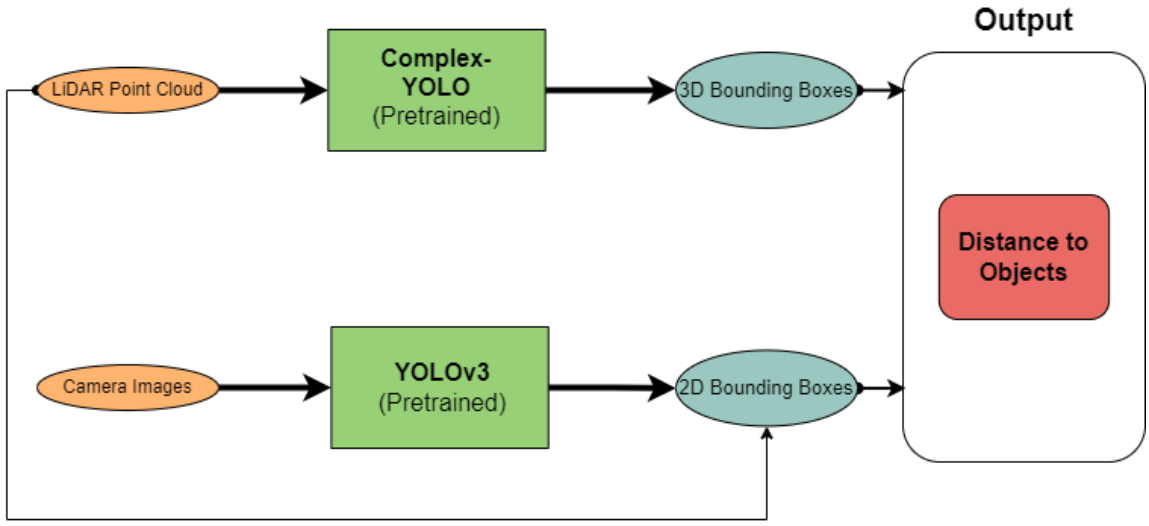

Figure 4: *This figure illustrates the finalized outcome of the project.*

Remark:

We will verify the models on a dataset different from KITTI dataset.

IV. References

[1] A. Bochkovskiy, C.-Y. Wang, en H.-Y. M. Liao, “YOLOv4: Optimal Speed and Accuracy of Object Detection”, *arXiv [cs.CV]*. 2020.

[2] S. Dholakia, “Perception: How self-driving cars 'see' The world,” *Medium*, 19-Mar-2019. [Online]. Available: https://swarit.medium.com/perception-how-self-driving-cars-see-the-world-ae630636f4c. [Accessed: 01-Feb-2022].

[3] A. Geiger, P. Lenz, en R. Urtasun, “Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite”, in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[4] V. K. Jonnalagadda, “Object detection yolo V1 , V2, V3,” *Medium*, 31-Jan-2019. [Online]. Available: https://medium.com/@venkatakrishna.jonnalagadda/object-detection-yolo-v1-v2-v3-c3d5eca2312a. [Accessed: 10-Feb-2022].

[5] S. A. S. Mohamed, M.-H. Haghbayan, T. Westerlund, J. Heikkonen, H. Tenhunen, en J. Plosila, “A Survey on Odometry for Autonomous Navigation Systems”, IEEE Access, vol 7, bll 97466–97486, 2019.

[6] S. S. Mohammadi., “Computer Vision for autonomous mobile robotics applications,” MSc Dissertation, Dept. of Signals and Systems., University of Chalmers, Goteborg, Sweden, 2011.

[7] S. D. Pendleton *et al.*, “Perception, Planning, Control, and Coordination for Autonomous Vehicles”, *Machines*, vol 5, no 1, 2017.

[8] O. Poliarus, Y. Poliakov, en A. Lebedynskyi, “Detection of Landmarks by Autonomous Mobile Robots Using Camera-Based Sensors in Outdoor Environments”, IEEE Sensors Journal, vol 21, no 10, bll 11443–11450, 2021.

[9] J. Redmon en A. Farhadi, “YOLO9000: Better, Faster, Stronger”, arXiv [cs.CV]. 2016.

[10] J. Redmon en A. Farhadi, “YOLOv3: An Incremental Improvement”, CVPR, vol abs/1804.02767, 2018.

[11] M. J. Shafiee, B. Chywl, F. Li, en A. Wong, “Fast YOLO: A Fast You Only Look Once System for Real-time Embedded Object Detection in Video”, CVP R, vol abs/1709.05943, 2017.

[12] A. Sharma, D. Morales, G. Kantor, en H. Choset, “Towards removing artificial landmarks for autonomous exploration in structured environments”, 2005.

[13] M. Simon, S. Milz, K. Amende, en H.-M. Gross, “Complex-YOLO: Real-time 3D Object Detection on Point Clouds”, CoRR, vol abs/1803.06199, 2018.