Real-Time 3-D Segmentation on An Autonomous Embedded System: using Point Cloud and Camera

Dewant Katare and Mohamed El-Sharkawy
IoT Collaboratory IUPUI, Department of Electrical and Computer Engineering
Purdue School of Engineering and Technology Indianapolis
dkatare@iupui.edu melshark@iupui.edu

***Abstract*—Present day autonomous vehicle relies on several sensor technologies for it's autonomous functionality. The sensors based on their type and mounted-location on the vehicle, can be categorized as: line of sight and non-line of sight sensors and are responsible for the different level of autonomy. These line of sight sensors are used for the execution of actions related to localization, object detection and the complete environment understanding. The surrounding or environment understanding for an autonomous vehicle can be achieved by segmentation. Several traditional and deep learning related techniques providing semantic segmentation for an input from camera is already available, however with the advancement in the computing processor, the progression is on developing the deep learning application replacing traditional methods. This paper presents an approach to combine the input of camera and lidar for semantic segmentation purpose. The proposed model for outdoor scene segmentation is based on the frustum pointnet, and ResNet which utilizes the 3d point cloud and camera input for the 3d bounding box prediction across the moving and non-moving object and thus finally recognizing and understanding the scenario at the point-cloud or pixel level. For real time application the model is deployed on the RTMaps framework with Bluebox (an embedded platform for autonomous vehicle). The proposed architecture is trained with the CITYScpaes and the KITTI dataset.**

***Index Terms*—Autonomous embedded platform, BLBX2, camera, CNN, LiDar, KITTI, point-cloud, RTMaps, segmentation.**

I. Introduction

Semantic image segmentation has seen significant improvement in the last two decades, In computer vision traditionally used methods for semantics segmentation of an image varies from local binary pattern [14], multi-scale detector [19], Histogram of oriented gradients [20], to Features from accelerated segment test [21]. The two widely used approach for semantic segmentation is supervised and unsupervised based. For unsupervised category the commonly used techniques are: thresholding, and K-means clustering, while for the supervised category the most common techniques are the conditional random field, markov random network, and support vector machines. These supervised models and their variations have been widely used recently and have resulted in the best performance. For vision related tasks in autonomous vehicle the significance is given to the combination of sensor values that can result in a function as an output for the execution of multiple tasks. As the autonomous vehicle is based on the sense, think and act model, substantial emphasis is on developing deep learning architecture for developing data driven statistics based efficient and accurate algorithms for replacing the traditional used methods. An autonomous vehicle flow comprises of Sensing, computing and actuation, the paper [5] focuses on the computing flow specifically on the perception module, As already mentioned the perception module relies on the inputs from the line of sight sensors such as camera's and lidar for the precise sensing of the surrounding, thus resulting in the tasks related to the localization, detection and prediction related to the surrounding of the vehicle. Recent researches proves that 3d point cloud from lidar implemented with deep learning architectures can be used for efficient detection and prediction [9]. Related to autonomous vehicle the semantic segmentation can be defined as acquiring the images or frames from the camera, or the point clouds from the lidar to observe and interpret the object at the pixel and point level and thus symbolize each of these label into substantial categories or classes such as traffic signs, lane, vehicle, pedestrians as shown in figure 1. In deep learning techniques convolution neural network has achieved exceptional results for pixel based semantic segmentation. The first significant paper [14] on the image segmentation based the fully convolution neural network introduced the idea of replacing the fully connected layers by the convolution layer. The paper also contains the use of interpolation layer in the architecture which determines that size of output (segmented image) is the same as the size of input. Another popular technique in the CNN involved use of deconvolution layer [24] for semantic segmentation, these deconvolution layer identifies the classes pixel-wise and finally predict the segmentation mask for these identified classes.

Another popular technique derived from traditionally used methods is the combination of deep convolutional neural network with the conditional random field [17], which reduces the imperfect localization of the feature for very precise and accurate object segmentation. For the 3d point cloud based deep learning Frustum-Pointnet [8] architecture is presented by Qi et al., which is capable of simultaneous segmentation and 3d bounding box prediction across the object. The model of the the architecture is based on three steps which are: frustum proposal, Segmentation and finally the bounding box estimation, this is discussed in detail in section III. Another framework, PointFusion [9] is proposed by Xu et al. which utilizes the Pointnet and ResNet [11] architecture for passing the frustum and 2D object detector respectively. These all mentioned deep learning has resulted in remarkable improvement for

This is the author's manuscript of the article published in final edited form as:

Katare, D., & El-Sharkawy, M. (2019). Real-Time 3-D Segmentation on An Autonomous Embedded System: Using Point Cloud and Camera. 2019 IEEE National Aerospace and Electronics Conference (NAECON), 356–361. https://doi.org/10.1109/NAECON46414.2019.9057988

Fig. 1. Image Semantic Segmentation [17]

the semantic image segmentation. However, the achieved high accuracy is a counterfeit with respect to its implementation on the autonomous embedded system. The primary objective of this paper is to propose an efficient framework implemented on the autonomous embedded platform which solves the problem of classification or object detection based on the semantic segmentation from the 3D lidar data by integrating supervised learning and deep learning of the local and global feature vector map from the 3d point clouds. As mentioned the proposed method utilizes the predicted 3d bounding box across the object and performing segmentation on the point features.

II. Related Works

Algorithms such as RANSAC [15] which is based on plane extraction from 3d point cloud were successful, however it required high computational power thus making them difficult for embedded system deployment. Recent research shows that 3d point cloud contains ample amount of the local features which can be extracted successfully, using convolutional neural network [22] [23]. The Squeezeseg [22] is based on the squeezenet [26] and is capable of real-time segmentation (as shown in figure 2) with the inclusion of traditional technique: conditional random field to improve the performance of the segmentation mask. Similarly, the PointSeg architecture is also based on squeezenet, however with an addition of transforming the raw 3d point cloud data into a a spherical image and then passing the transformed data to the architecture. This transformed data contains global and local feature which can be extracted by the CNN. The CNN architecture comprised of the fire layer, squeeze reweighting layer and enlargement layer.

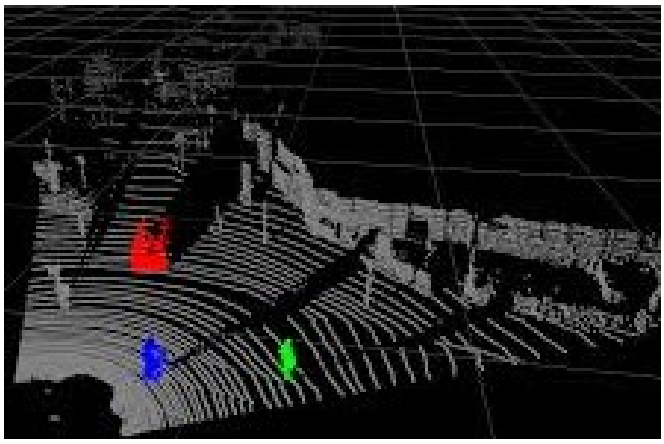

Fig. 2. An example of 3D object segmentation [22]

III. 3D Point cloud Segmentation

The 3d point cloud segmentation is based on the classification and localization of the points in the 3d plane. The point or pixel wise depth related information of the object present in the region can be represented in the form of point cloud in the 3d coordinate. A 3d frustum of the region can be obtained in the form of the projection matrix. If the camera is integrated with lidar, then the deep learning algorithm requires the image with depth or the lidar point cloud as an input, and outputs the 3D segmentation for the present objects in the line of sight. The integrated camera provides the 2d image region and the lidar provides the point cloud through which a frustum can be proposed which acts as plane to look for the local and global feature of the present objects in that frustum.

Proposed Model: The proposed model utilizes the frustum point cloud as an input and performs the binary classification for the points available in the 3d plane, thus predicting whether a particular point belong to the object, the output is in the form of the probability score. The proposed model includes an additional feature (figure 6) exploiting the global feature from the lidar and output feature of the image from the ResNet architecture. As shown in figure 6 the lidar global feature vector is passed as an input to a fully connected network with five layer which outputs the centroid position of the 3D bounding box across the object which is binary classified on the previous step. Also, the global feature from Lidar is also connected as an input to the image feature vector, which is then passed to the another fully connected network with five layers that outputs the value corresponding to the position estimation of 3D bounding box (length, width, height and angle). The image feature comprising of 2D bounding box

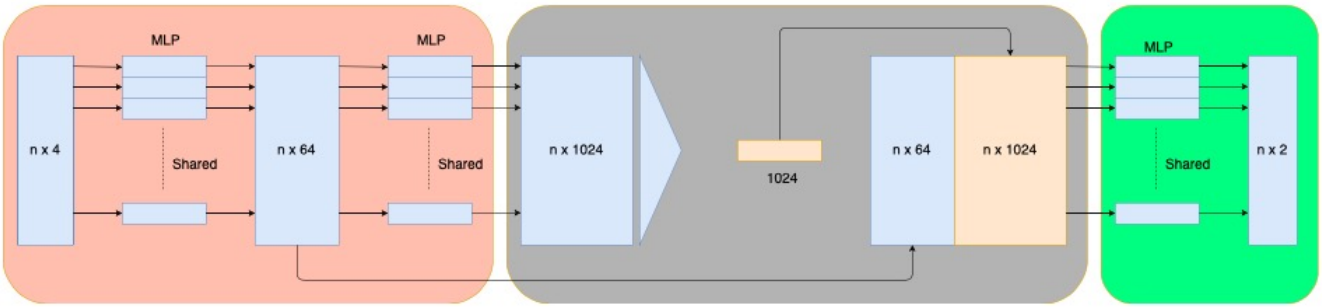

Fig. 3. Instance Segmentation PointNet

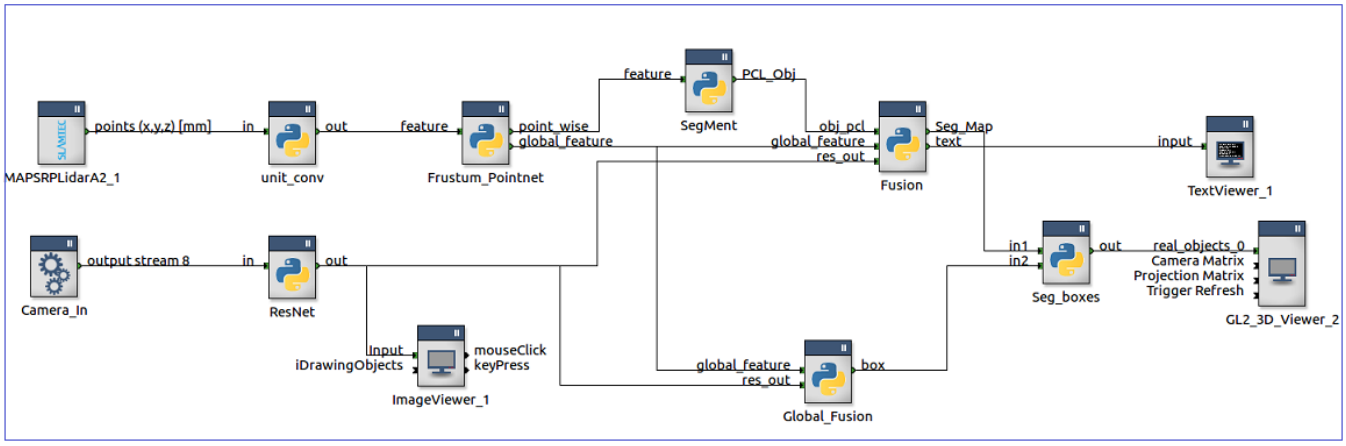

Fig. 4. Proposed model for Bluebox 2.0 in the RTMaps framework for Real time segmentation and bounding box prediction

described above, is the output of the 2D object detection or classification from the ResNet architecture.

Instance Segmentation Pointnet: The Instance Segmentation pointnet is the main building block of the architecture that uses the extracted frustum point cloud as an input and provides two classification scores for each of the n input points as an output. As mentioned previously, the output score is the measure of the predicted probability which describes if the related point belongs to the chosen object. This network is a modified or upgraded version of Pointnet [6]. Very likely to the 2d image segmentation the feature or the points from another object may overlap with another object, in this case the predicted 3D bounding box for each object plays a vital role as it separates them. The network architecture is schematically illustrated in Figure 3.

T-Net: This T-Net proposes the centroid position (x,y,z) of the object from the point cloud, for the 3D bounding box position. The input to T-net is the object point cloud but it only utilizes the 3D coordinate points of the each given points. The block diagram is shown in figure 5.

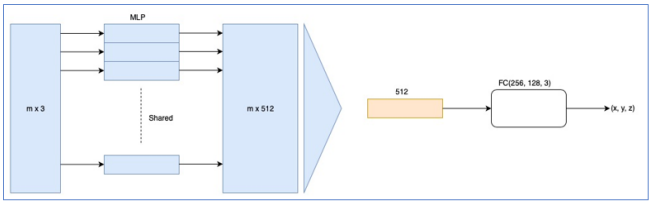

Fig. 5. T-net architecture

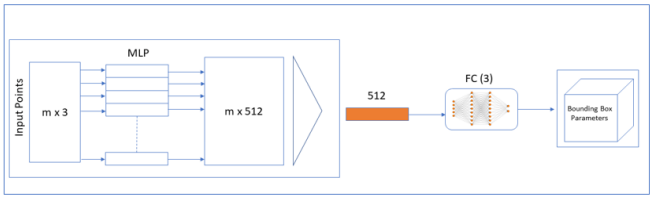

Fig. 6. Bounding box estimation block

IV. Implementation

The model is implemented in Python programming language using the PyTorch framework [1] [16] and point cloud library as the modules can be easily implemented in autonomous embedded platform using Arm Ubuntu OS. In the architecture, the multilayer perceptron layers are implemented using the convolution module (nn.Conv1d) from pytorch. The points in the frustum point-cloud which lies inside the labelled 3D bounding boxes is recognized as the part of the object.

This is the similar approach followed in Frustum-pointnet [8] to use the ground truths for the Instance segmentation block. For accessing the Image feature vector ResNet architecture comprising of 34 layers is implemented with torchvision [1][16]. The ResNet architecture uses pre-trained model for the initialization of the weights. The pre-trained model inputs image of size (224 , 224) with 3 channel.

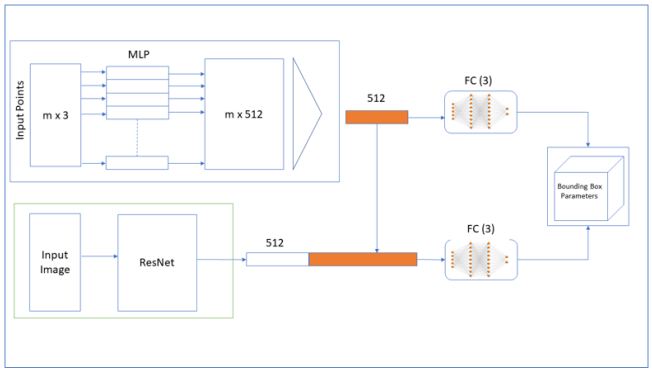

Fig. 7. Proposed Lidar and Camera fused bounding box estimation Segmentation with Bounding Box

Loss Function: The loss function for the proposed model can be described as follows:

$$l = l_{isp} + (l_{t-net} + l_{box-center} + 0.1 l_{corner} + l_{box-size} + l_{clf})$$

here l_{isp} is the log loss corresponding to the output of Instance Segmentation pointnet, $l_{t\text{-}net}$ corresponds to the log loss for the 3D box centroid in the T-Net, $l_{box\text{-}center}$ corresponds to the log loss of the 3D box center for the added model into the existing frustum-pointnet architecture, l_{corner} is corner loss of the bounding box, $l_{box\text{-}size}$ corresponds to the predicted box size regression, l_{clf} corresponds to the log loss of the classification output.

Data Augmentation: One of the most important factor to consider during the training of an architecture is the right learning of the point clouds, which corresponds to how accurately an architectures predicted the value correctly matching to the ground truth values. This situation generally occurs when the architecture or model learns the noise feature instead of the signal features. For the proposed model the overfitting is avoided by randomly rotating the labelled data from 3D bounding box and points in the frustum point cloud, by the angle uniformly sampled around the camera's Y-axis, and for the frustum point cloud it is augmented by random sampling a subset of points in the frustum point cloud.

Optimizer: For optimization of the proposed model, the Adam optimizer is used. the adam optimizer follows an adaptive learning rate by using the first and second order moment of the gradient to update the weights. The starting learning rate of 0.001 is chosen as random value along with the batch size of 32 frustum point clouds. After every 50 epochs the learning rate is reduced by half of it's previous value. The loss is calculated for the complete validation set on the completion of an epoch, the training is stopped once the validation loss shows constant values without major change for 80-100 epochs.

V. Autonomous Embedded System

For a system with autonomous functionality, the important part is integration between the hardware and software module [18]. As already mentioned the proposed model from hardware perspective is based on lidar and camera sensor and from software perspective relies mainly on python programming language. This section provides information for the implementation of Segmentation application on the autonomous embedded system (Bluebox 2.0) and clarifies its integration with RTMaps embedded Software.

A. Bluebox

It is an ARM processor based development platform for autonomous vehicles. The development platform consists of three independent systems from automotive processor family: LS2084A (a high computational unit), S32V234 (vision processor and sensor fusion board) and S32R274 which is a radar based micro-controller.

The software for the LS2084A and S32V234 consists of ARM based Linux board support package which is built using the Yocto framework. The LS2084A and S32V234 processors are installed with ARM-Ubuntu 16.04 LTS which is a fully developer-supported system and contains the kernel source code, toolchains, compilers, ROS and Docker package [2]. The hardware and software overview is shown in Figure 8.

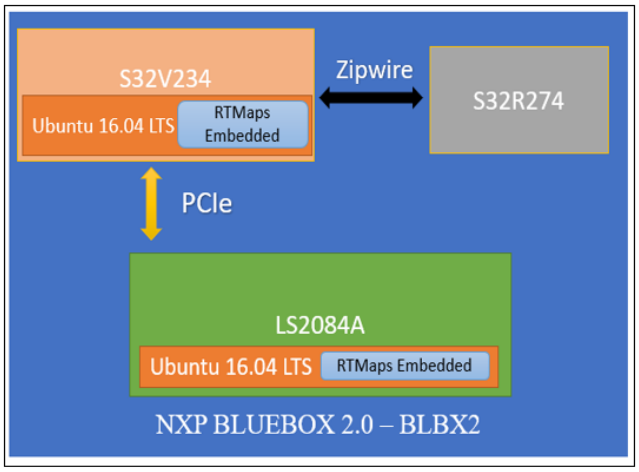

Fig. 8. System overview of NXP BLBX2 with RTMaps

B. RTMaps

RTMaps is a graphical supported design based software framework designed for multisensor based multimodal applications. It has been tested and used previously for acquisition, processing and fusing the data streams in the real-time or even in the post-processing scenarios. The software architecture consists of several independent modules that can be used

for different situation and circumstance [10] [13] [18], the important modules used for the proposed model are mentioned below.

RTMaps Component Library: : The RTMaps component library comprises of software module developed using RTMaps Software develpoment kit comprising of C++ and Python. The component libraries are used for several functions such as communication, data conversion, perception, sensor fusion, detection, 2D - 3D display and localization which are responsible for the development of an application [10] [13] [18].

RTMaps Embedded: : RTMaps embedded is highly optimized version of RTMaps Studio software package and comprises of the component library and the RTMaps runtime engine with the capability of running on an embedded x86 or ARM capable platform such as NXP Bluebox, DSpace MicroAutobox [10] [13] [18].

For the model deployment RTMaps embedded platform is used. It is used independently on the Bluebox platform to run the proposed model, and with the RTMaps remote studio operating on a Desktop computer to provide the graphical interface for the visualization of the segmented object and testing purpose. The connection between the Computer running RTMaps studio and the Bluebox Embedded platform is accessed via a static TCP/IP as shown in Figure 8.

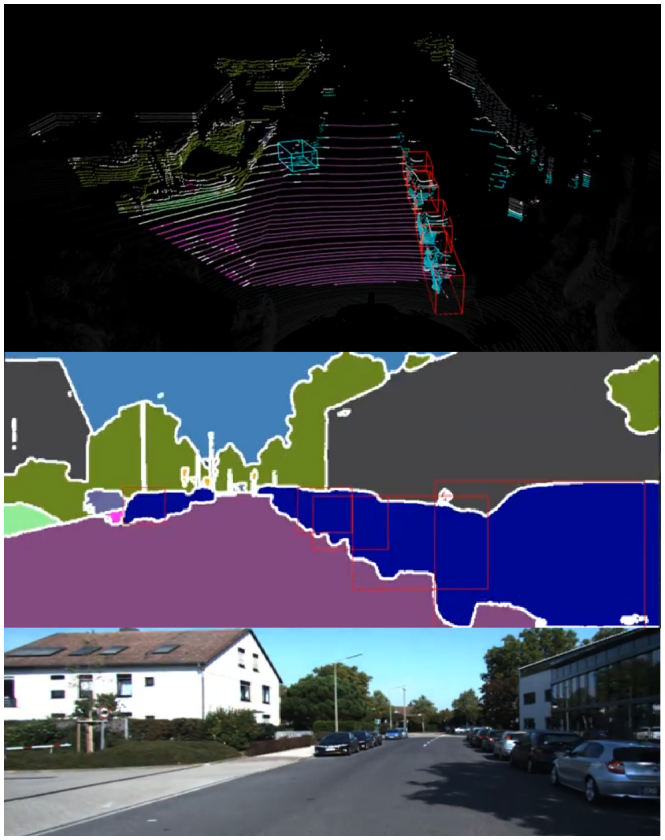

Fig. 9. Output from the Segmentation and the 3D bounding boxes on the KITTI dataset

VI. Results

The proposed model is trained with CITYscapes, KITTI train and test dataset [3] and evaluated on KITTI val dataset. This section presents the results for the trained and tested data. It can be seen from table I that the proposed model with the Frustum PointNet implementation has performance close to the baseline model. Figure 9 shows the predicted segmentation of the vehicle class along with the enclosed 3D bounding boxes from the lidar point cloud, Segmentation from the image point of view and finally the original view from the camera view. Figure 10 and figure 11 shows the validation accuracy and precision score for the KITTI dataset. After 250 - 310 epochs the architecture outputs constant accuracy and precision score with minor change, and relatively does not change after 350-390 epochs. The proposed model performs well in most of the scenarios, however when the view point is on the turning or edges of the scenario it predicts false bounding boxes thus mixed classes for the segmentation, because of the false angle assumption. The results with respect to the KITTI dataset are shown in table I.

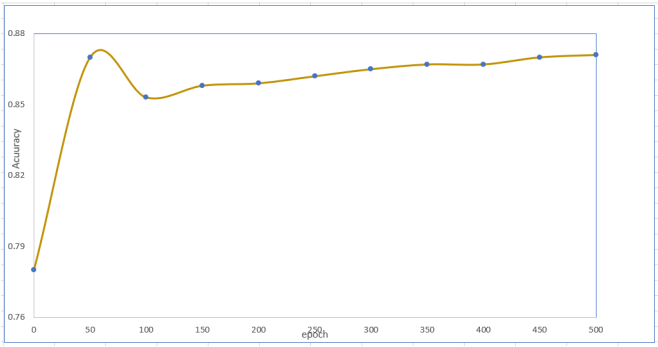

Fig. 10. Validation accuracy curve on KITTI dataset

As mentioned in previous section, the RTMaps embedded framework consists of several software module which can be used to collect the sensor data. The Figure 4 shows the proposed model based RTMaps diagram, using Lidar and camera as an input and processing it through the proposed architecture for the segmentation and the 3D bounding box prediction across the object. For the general use of the proposed model the lidar module shown in figure 4 can be replaced with any CAN-Frame supported lidar module [5] available. If the CAN-Frame component module is used to acquire the sensor data then it can be further processed from the company specific lidar module which splits the can bus information into: desired parameters such as raw point clouds or x,y,z dimensions and yaw angle, using the vendor specific lidar can bus splitter module. These parameter or sensed inputs is further passed through the proposed algorithm which then based on the input scenario predicts the 3D bounding boxes.

VII. Conclusion

This paper proposes an approach to combine the sensed inputs from the Lidar and camera and then process them

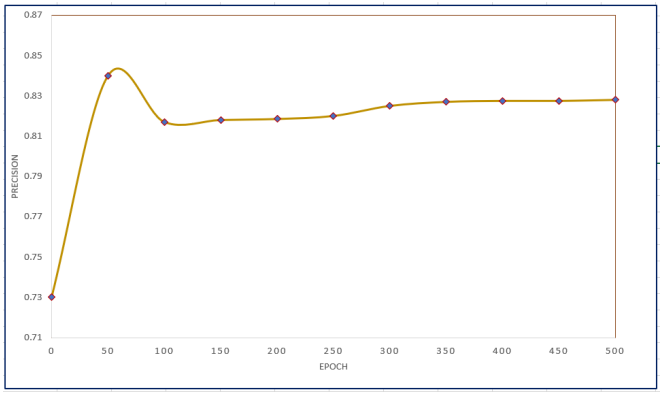

Fig. 11. Validation precision score curve on KITTI dataset

TABLE I
OUTPUT OF KITT DATASET FOR VEHICLE ON BASELINE AND MODIFIED ARCHITECTURE

Method	Precision	Recall
Frustum-Pointnet	76.4	90.92
Proposed Frustum-Pointnet - KITTI	74.80	91.83
Proposed Frustum-Pointnet - KITTI (50%)	81.76	92.86

through Frustum-pointnet architecture with the purpose of Segmentation of object present in the line of sight. Here we propose the Segmentation of the identified point cloud on the basis of defined classes and finally the prediction of 3D bounding boxes across the detected object from the point clouds frustum. The proposed detection model is designed in the RTMaps frame-work, to avail the functionality of real-time detection. The proposed method can be divided into two parts: First, a Lidar and camera based model implemented from the RTMaps component library which is capable of acquiring the real-time sensor values and Secondly, a deep learning algorithm implemented in the python module of the RTMaps components as shown in Figure 4, which on the basis of the input values from the lidar and camera classifies, segment and predict the bounding boxes on the detected vehicle.

REFERENCES

[1] Paszke, Adam, et al. "Automatic differentiation in pytorch." (2017).
[2] NXP Semiconductors,https://www.nxp.com/applications/solutions/ automotive accessed: 2019-03-14.
[3] Geiger, Andreas, Philip Lenz, and Raquel Urtasun. "Are we ready for autonomous driving? the kitti vision benchmark suite." 2012 IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2012.
[4] Verbickas, Rytis, et al. "SqueezeMap: fast pedestrian detection on a low-power automotive processor using efficient convolutional neural networks." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. 2017.
[5] S. Kato, S. Tokunaga, Autoware on board: enabling autonomous vehicles with embedded systems, 9th ACM/IEEE Intenational conference on cyber-physical systems, Portugal 2018.
[6] Qi, Charles R., et al. "Pointnet: Deep learning on point sets for 3d classification and segmentation." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017.
[7] Zhou, Yin, and Oncel Tuzel. "Voxelnet: End-to-end learning for point cloud based 3d object detection." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018.
[8] Qi, Charles R., et al. "Frustum pointnets for 3d object detection from rgb-d data." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018.
[9] Xu, Danfei, Dragomir Anguelov, and Ashesh Jain. "Pointfusion: Deep sensor fusion for 3d bounding box estimation." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018.
[10] Intempora.com. (2018). Intempora - RTMaps: For multi-sensor applications. https://intempora.com/prducts/rtmaps accessed: 2019-03-14
[11] He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.
[12] Mousavian, Arsalan, et al. "3d bounding box estimation using deep learning and geometry." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017.
[13] Katare, Dewant, and Mohamed El-Sharkawy. "Embedded System Enabled Vehicle Collision Detection: An ANN Classifier." 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC). IEEE, 2019.
[14] Darrell, JLaESaT, and U. C. Berkeley. "Fully Convolutional Networks for Semantic Segmentation." UC Berkeley (2014).
[15] Douillard, Bertrand, et al. "On the segmentation of 3D LIDAR point clouds." 2011 IEEE International Conference on Robotics and Automation. IEEE, 2011.
[16] PyTorch, https://pytorch.org/ accessed: 2019-03-14.
[17] Chen, Liang-Chieh, et al. "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs." IEEE transactions on pattern analysis and machine intelligence 40.4 (2018): 834-848.
[18] Katare, Dewant, and Mohamed El-Sharkawy. "Collision warning system: embedded enabled (RTMaps with NXP BLBX2)." 2018 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT). IEEE, 2018.
[19] Leutenegger, Stefan, Margarita Chli, and Roland Siegwart. "BRISK: Binary robust invariant scalable keypoints." 2011 IEEE international conference on computer vision (ICCV). Ieee, 2011.
[20] Dalal, Navneet, and Bill Triggs. "Histograms of oriented gradients for human detection." international Conference on computer vision Pattern Recognition (CVPR'05). Vol. 1. IEEE Computer Society, 2005.
[21] Rosten, Edward, and Tom Drummond. "Fusing points and lines for high performance tracking." ICCV. Vol. 2. 2005.
[22] Wu, Bichen, et al. "Squeezeseg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3d lidar point cloud." 2018 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2018.
[23] Wang, Yuan, et al. "Pointseg: Real-time semantic segmentation based on 3d lidar point cloud." arXiv preprint arXiv:1807.06288 (2018).
[24] Noh, Hyeonwoo, Seunghoon Hong, and Bohyung Han. "Learning deconvolution network for semantic segmentation." Proceedings of the IEEE international conference on computer vision. 2015.
[25] Iandola, Forrest N., et al. "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and¡ 0.5 MB model size." arXiv preprint arXiv:1602.07360 (2016).
[26] Chen, Xiaozhi, et al. "Monocular 3d object detection for autonomous driving." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016.
[27] Zheng, Shuai, et al. "Conditional random fields as recurrent neural networks." Proceedings of the IEEE international conference on computer vision. 2015.
[28] Long, Jonathan, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.
[29] Liu, Wei, et al. "Ssd: Single shot multibox detector." European conference on computer vision. Springer, Cham, 2016.