# Real-time Spatial-temporal Context Approach for 3D Object Detection using LiDAR

K. S. Chidanand Kumar[1] and Samir Al-Stouhi[2]
[1]*Great Wall of Motors, Whitefield, Bangalore, Karnataka, India*
[2]*American Haval Motors, Michigan, U.S.A.*

Keywords: Bird's-Eye-View (BEV), Convolutional Neural Network (CNN), Non-Local Context Network (NLCN), YOLO, Convolutional LSTM (CLSTM), Spatial-Temporal Context Network (STCN).

Abstract: This paper proposes a real-time spatial-temporal context approach for BEV object detection and classification using LiDAR point-clouds. Current state-of-art BEV object-detection approaches focused mainly on single-frame point-clouds while the temporal factor is rarely exploited. In current approach, we aggregate 3D LiDAR point clouds over time to produce a 4D tensor, which is then fed to a one-shot fully convolutional detector to predict oriented 3D object bounding-box information along with object class. Four different techniques are evaluated to incorporate the temporal dimension; a) joint training b) CLSTM c) non-local context network (NLCN) d) spatial-temporal context network (STCN). The experiments are conducted on large-scale Argoverse dataset and results shows that by using NLCN and STCN, mAP accuracy is increased by a large margin over single frame 3D object detector and YOLO4D 3D object detection with our approach running at a speed of 28fps.



## 1 INTRODUCTION

An autonomous vehicle is an intelligent transportation which must operate safely, accurately observe its environment to make robust decisions and navigate in a complex traffic environment. A typical autonomous system is divided into subtasks (J. Levinson et al., 2011) perception, prediction, planning and control. Perception is in charge of estimating all actor's positions and motions, given the current and past evidences. Prediction on the other hand, tackles the problem of estimating the future positions of all actors as well as their intentions (e.g., changing lanes, parking). Finally, motion planning takes the output from previous stacks and generates a safe trajectory for the self-driving vehicle to execute via a control system.

3D object detection is a fundamental task in perception systems. Recent approaches to 3D object detection exploit different data sources. Camera based approaches utilize either monocular (X. Chen et al., 2016) or stereo images monocular (X. Chen et al., 2017), fisheye cameras or depth cameras. These camera-based approaches have drawbacks such as limited fields of view, difficult in operating under low-contrast conditions and inability to determine precise distances within the surrounding outdoor environment. On the other hand, LiDAR sensors, which use reflected laser pulses to scan the area around a vehicle, can overcome such limitations. LiDAR scanner data is used to create a 360-degree point cloud, which solves the limited field of view problem experienced in camera-based systems, and LiDAR data is more robust to changes in weather and illumination issues in indoor and outdoor environments. Thus, they are generally considered as more important sensors than cameras for autonomous vehicles driving safety and are adopted by nearly all auto-makers today (Baidu 2017, Google's waymo 2017, What it Was Like 2017 and Volvo 2018).

Compared to images, Lidar point clouds are sparse with a varying density, highly unordered, noisy due to imperfect reflections and echoes. Also LiDAR point cloud lack colour and texture features that characterize the object classes as in the case of 2D camera perspective images. Such complexity, in addition to the dynamic nature of the environment, motivates us to incorporate the temporal factor in addition to the spatial features of the input 3D LiDAR point clouds.

Real-time performance is much essential in autonomous driving systems. While deep-learning



Kumar, K. and Al-Stouhi, S.
Real-time Spatial-temporal Context Approach for 3D Object Detection using LiDAR.
DOI: 10.5220/0009340004320439
In *Proceedings of the 6th International Conference on Vehicle Technology and Intelligent Transport Systems (VEHITS 2020)*, pages 432-439
ISBN: 978-989-758-419-0

has a well-known success story in camera-based computer vision. In this context, literature survey tackles the problem of real-time performance using Single shot detectors, like YOLO (Redmon, J et al., 2016) and SSD (Liu, W et al.,2016).

3D object detection using LiDAR point clouds are mainly divided into two types: 3D voxel grids and 2D projections. A 3D voxel grid transforms the point cloud into a regularly spaced 3D grid called voxels, and from each voxel cell we can compute statistics and apply 3D convolutions to extract high-order representation from the voxel grid (M. Engelcke et al., 2017). However, point clouds are sparse by nature, the voxel grid are also sparse, less compact and require huge computation. As a result, typical systems ((M. Engelcke et al., 2017, B. Li,) et al., 2016) only run at 1-2 FPS. On the other hand, 2D projection based techniques projects the point cloud onto a plane, which is then discretized into a 2D image based representation where 2D convolutions are applied. These 2D projection based representations are more compact, but they bring information loss during projection and discretization. In addition to computation efficiency, BEV representation also has other advantages i.e. it eases the problem of object detection as objects do not overlap with each other and thus the network can exploit priors about the physical dimensions of objects.

In our framework, we use single-shot detection based architecture to detect objects on LiDAR's BEV.

## 1.1 Related Work

Most of the works on 3D object detection using LIDAR BEV representation relies on single point cloud PIXOR (Yang, B et al., 2018), Complex YOLO (M. Simon et al., 2018), YOLO3D (Ali, W et al., 2018). These LiDAR 3D point clouds object detection methods do not take the advantage of temporal information to produce more accurate 3D bounding boxes. Recently, Fast and Furious (Luo W et al.,2018), IntentNet (S. Casas et al., 2018), Neural motion planner (Wenyuan Zeng et al., 2019) incorporates the time with 3D voxels using 2D, 3D convolutions and adopts a multi-task learning like tracking, motion forecasting and motion planning. To our knowledge YOLO4D (El Sallab., 2018) is the only technique where they incorporated temporal information from successive point clouds.

In this paper, we exploit temporal information from successive point clouds using spatial-temporal model to augment context information for BEV based 3D object detection.

## 1.2 Contribution

In this paper, we propose a spatial-temporal context based 3D object detector that operates on sequence of 3D point clouds. Our approach is a single-stage 3D object detector that exploits the 2D BEV representation in an efficient way since it is computationally less expensive as compared with 3D voxel grids, and also preserves the metric space which allows our model to explore priors about the size and shape of the object categories. Our detector outputs accurate oriented bounding boxes in real-world dimension in bird's eye view. The main contributions of this paper are:

a) Non-local context network (NLCN), a novel approach to augment the CNN backbone features for BEV object detection by a context representation computed using non-local relations between feature maps to capture global appearance and motion information. This approach has led to significant improvements by 4.4mAP over single-frame BEV object detector and by 1.1mAP over YOLO4D BEV object detector on Argoverse dataset (M. Chang et al., 2019).

b) Spatial-temporal context network (STCN), a novel approach of generating context representation for BEV object detector by applying 2D convolutions on a stack of BEV images (Super image) to capture local spatial-temporal information and using 3D convolutions on local spatial-temporal feature maps to capture global temporal interactions (long-range temporal dynamics). This approach led to significant improvements by 6.9mAP over single-frame 3D object detector and by 3.5mAP over YOLO4D 3D object detector on Argoverse dataset (M. Chang et al., 2019).

The rest of the paper is organized as follows; first, we discuss the single frame based 3D object detection, followed by the spatial-temporal approaches to encode context information from temporal point cloud sequences. Finally, we present our experimental results and evaluate different techniques on Argoverse dataset (M. Chang et al., 2019).

# 2 SPATIO-TEMPORAL 3D OBJECT DETECTION

In this section, the approach for spatial-temporal BEV object detection is described. The main motivation behind our work is to exploit not only the spatial but also the temporal information in the input LIDAR

sequences for more accurate object detection. For encoding temporal sequences, we experimented with four different approaches. These approaches model the temporal information in different ways.

As shown in Fig. (1), BEV maps are generated from a LIDAR point-clouds(PC's) and each BEV maps were given to CNN backbone network (a combination of few convolutional and maxpool layers) to extract feature maps. To encode long-range temporal information, successive BEV maps were given to context generation block which employs four different approaches to encode temporal information. Backbone feature maps are concatenated with spatial-temporal feature maps and it will be fed to header-network which consists of fewer convolutional layers followed with classification and regression branches to handle both object recognition and localization.

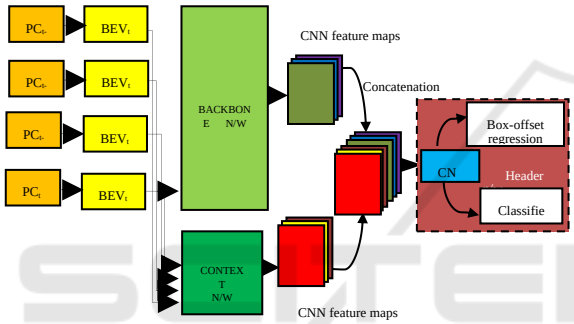


Figure 1: Spatial-temporal 3D object detection.

## 2.1 Input Representation

3D point clouds are highly unstructured, and thus standard convolutions cannot be directly applied since they assume that the input lies on a grid. One option is to use voxelization to form a 3D voxel grid and then we can use 3D convolution to extract 3D feature. However, this can be very expensive in computation as we have to slide the 3D convolution kernel along three dimensions. Instead, we can represent the scene from the BEV alone.

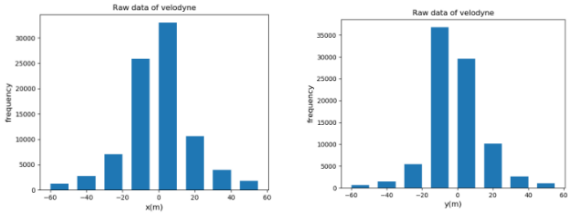


Figure 2: Velodyne FOV range estimation in X and Y directions on argoverse datasets.

By reducing the degrees of freedom from three to two, we don't lose information in point cloud as we can still keep the height information as channels along the third dimension.

In addition to computation efficiency, BEV representation also has other advantages. It eases the problem of object detection as objects do not overlap with each other (compared with front-view representation) and thus the network can exploit priors about the physical dimensions of objects.

In current approach, we estimate the velodyne FOV range for 3D object detection in BEV on argoverse dataset based on statistics of graphs shown in Fig. (2).

$$P_{\lambda\_argo} = \begin{Bmatrix} P = [x, y, z]^T, \\ \forall x \in [-51.2m, 51.2m], \\ y \in [-51.2m, 51.2m], \\ z \in [-2.1m, 1.5m] \end{Bmatrix} \quad (1)$$

We follow the design put by [13] to get single birds-eye-view RGB-map.

## 2.2 Single Frame 3D Object Detection

Single-frame 3D object detection network is a one-shot fully convolutional detector which mainly consists of backbone network for feature extraction and a header network for object recognition and localization as shown in Fig. (3). In our framework, we combined classification-bounding box prediction directly by predicting objects in each cell of the feature maps and the corrections on anchor boxes.

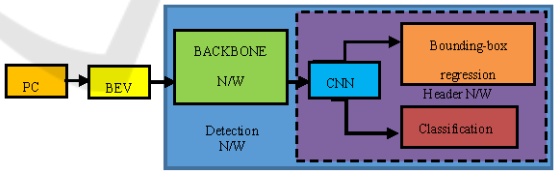


Figure 3: Single point-cloud based object detection framework.

Network details for single point-cloud based object detection is as shown in Fig. (4).

### 2.2.1 Backbone Network

Backbone network consists of seven convolutional blocks, and each conv2D layers with filter number {16, 32, 64, 128, 256, 512 and 1024}, kernel filter sizes of 3x3 and stride 1. After each of the first six convolutional blocks, a maxpool layer with kernel filter size 2x2 and stride 2 is incorporated. Multi-scale

features are generated by resizing and concatenating feature maps from different scales. The total down sampling rate of the network is 32.

### 2.2.2 Header Network

The header network is a multi-task network that handles both object recognition and localization. Similar to feature pyramid network (T.-Y. Lin et al., 2017), we predict oriented bounding boxes at two scales as shown in Fig. (4). At each scale we use three anchors at each location with predefined sizes, aspect ratios, and orientations. Anchors are calculated by taking the mean 3D box dimensions for each object class in argoverse dataset, and use these average box dimensions as our anchors.

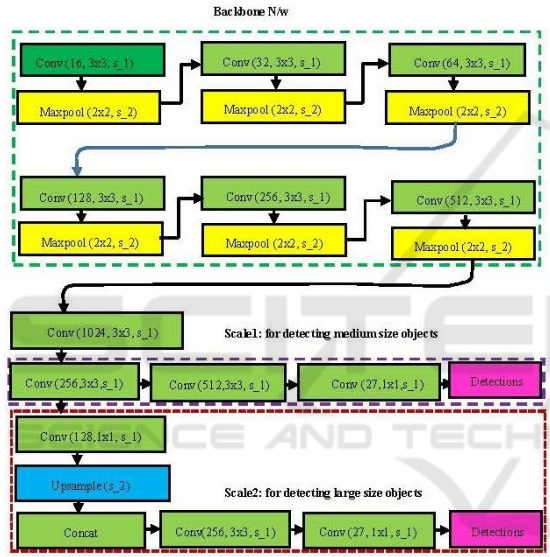


Figure 4: Architecture of single frame based object detection framework.

From the backbone network, we add few more CNN layers and only the last CNN layers predicts a 3D-tensor encoding oriented bounding-boxes, one confidence score and $N$ classes thus producing a tensor of size $MxMx[3x(4+2+1+N)]$ where $M$ is the feature map size. The classification branch outputs a score for each anchor indicating the probability of a vehicle at each anchor's location associated scale. The regression branch predicts regression targets $(t_x, t_y, t_w, t_h, \cos(b_\phi), \sin(b_\phi))$ for each anchor associated scale.

### 2.2.3 Loss-function Calculation

The loss function is similar to complex-YOLO (M. Simon et al., 2018) which consists of two parts. The first part of the loss function is simply a sum of squared errors similar to YOLO 2D (Redmon, J et al., 2016), while the second part is built on the Euler regression logic and is defined to be the difference between the complex numbers of prediction and ground truth.

$$L_{total} = L_{YOLO} + L_{Euler}$$

$$L_{Euler} = \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} L_{obj}[(t_{\Re} - \hat{t_{\Re}})^2 + (t_{\Im} - \hat{t_{\Im}})^2]$$

$$L_{YOLO} = \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} L_{ij}^{obj} \left[ \left( x_i - \hat{x_i} \right)^2 + \left( y_i - \hat{y_i} \right)^2 \right]$$
$$+ \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} L_{ij}^{obj} \left[ \left( \sqrt{w_i} - \sqrt{\hat{w_i}} \right)^2 + \left( \sqrt{h_i} - \sqrt{\hat{h_i}} \right)^2 \right] \quad (2)$$
$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} L_{ij}^{obj} \left[ \left( C_i - \hat{C_i} \right)^2 \right]$$
$$+ \sum_{i=0}^{S^2} L_i^{obj} \sum_{c \in classes} \left( p_i(c) - \hat{p_i}(c) \right)^2$$

## 2.3 Multi-frame 3D Object Detection

In this section, the approach for spatial-temporal 3D object detection is described. The main intuition behind our work is to leverage not only the spatial but also the temporal information in LiDAR input sequences for more accurate object detection. For encoding temporal sequences, we adopted four different approaches: joint training, CLSTM (YOLO4D), NLCN and STCN. These approaches encode the temporal information in different ways.

### 2.3.1 Joint Training

In this technique, point-cloud frames are jointly trained with single-shot fully convolutional detector.

Each BEV maps are processed through single-frame object detection network but total loss is combined on the last stage and it is computed as given in Equation. (3). Here the network weights of CNN backbone network on each BEV maps are shared during training thus reducing number of learnable parameters. During the training process, it is up to the network to learn the temporal information from the input joint training scheme without encoding hidden state through recurrent layers.

$$L_{total} = (L_{YOLO} + L_{Euler})_1 + (L_{YOLO} + L_{Euler})_2 + (L_{YOLO} + L_{Euler})_3 + (L_{YOLO} + L_{Euler})_4 \quad (3)$$

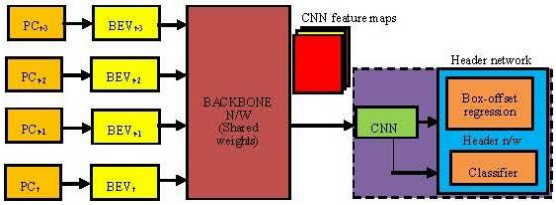


Figure 5: Joint training of successive point-clouds.

Architecture for joint training on successive point-clouds is as shown below in Figure (3).

### 2.3.2 Temporal Aggregation using CLSTM

In this architecture (El Sallab., 2018), a CLSTM (S. Xingjian et al., 2015) layer is injected directly into single-frame object detection architecture between the feature-extraction stage and header-network. CLSTM allows the network to learn both spatial and temporal information thus enhancing context information for 3D object detection. The network is trained on the successive point-clouds, thereby leading to a model that is capable of detecting objects in temporal streams of input point-clouds. This architecture maps an input frame $I$ and the previous state $s_{t-1}$ to a list of oriented bounding boxes D, and current state $s_t$ as shown in Equation. (4).

$$(I_t, s_{t-1}) \xrightarrow{yields} (D_t, s_t) \quad (4)$$

Where the state $s_t$ is used as input for the next time step predictions. The loss in this case is the same as in Equation. (2), however, the optimization is back-propagated through time via the injected CLSTM layer to maintain the temporal information.

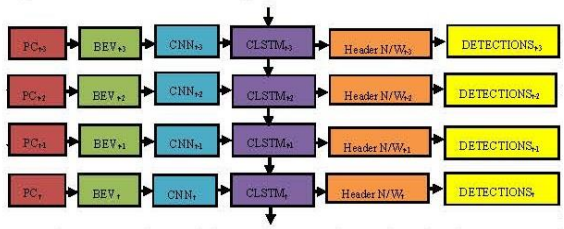


Figure 6: Joint training of successive point-clouds.

### 2.3.3 Temporal Aggregation using Non-Local Context Network (NLCN)

Usage of non-local layer (Xiaolong Wang et al., 2018) is commonly used in video action recognition but it is rarely exploited in BEV object detection. Hence to embed temporal characteristics on BEV maps, the non-local layer is introduced into the context CNN block which is as shown in Figure (7). The most distinguishing part of non-local neural networks is that it captures global dependencies by exploiting both appearance and motion features which has a significant impact in static/dynamic object detection. Given an input feature tensor $X \in \Re^{CxNxHxW}$ obtained from a sequence of $N$ feature maps of size $CxHxW$, we desire to exchange information between features across all spatial locations and frames.

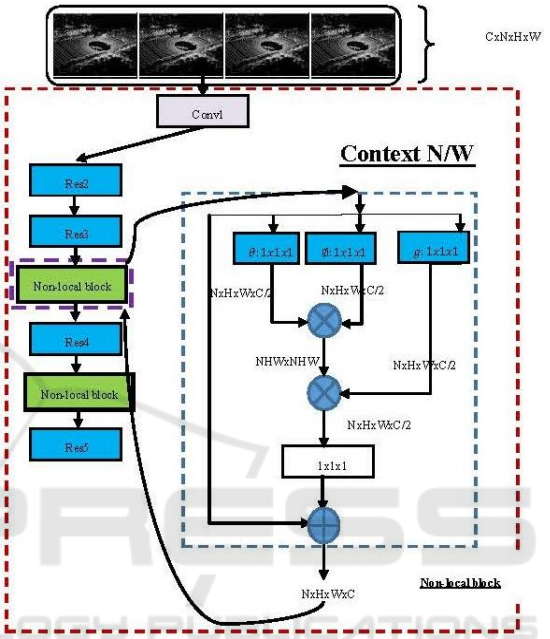


Figure 7: Non-local context network (NLCN).

Let $x_i \in R^C$ sampled from $X$, the corresponding output $y_i \in R^C$ of non-local operation can be formulated as follow:

$$y_i = \frac{1}{\sum_{\forall j} e^{\theta(x_i)^T \emptyset(x_j)}} \sum_{\forall j} e^{\theta(x_i)^T \emptyset(x_j)} g(x_j) \quad (5)$$

Here, $i, j = [1, NHW]$ indexes all locations across a feature map and all frames. We first project $x$ to a lower dimensional embedding space $R^{C'}$ by using linear transformation functions $\theta, \emptyset, g$ ($1 \times 1 \times 1$ convolution). Then, the response of each location $x_i$ is computed by the weighted average of all positions $x_j$ by using Embedded Gaussian instantiation. The overall non-local layer is finally formulated as $Z = W_Z$Y+X, where the output of nonlocal operation is added to the original feature tensor $X$ with a transformation $W_Z$ (1×1×1convolution) that maps $Y$ to the original feature space $R^C$. The intuition behind the non-local operation is that when extracting

features at a specific location in a specific time, the network should consider the spatial and temporal dependency within a sequence by attending on the non-local context.

In our architecture, we embed two non-local layers after the Residual 3rd and 4th blocks of SE-ResNext (J. Hu et al., 2017) module. In the training phase, we don't initialize the weights of SE-ResNext 2D convolution model from the ImageNet pre-trained model.

#### 2.3.4 Temporal Aggregation using Spatial-Temporal Context Network (STCN)

To model long range temporal dynamics, we generate a super-image by stacking $N$ BEV frames in the channel dimension to form a tensor of size $1x3NxHxW$. This super-image not only contains local spatial appearance information represented by individual point-cloud but also local temporal dependency among these successive video frames.

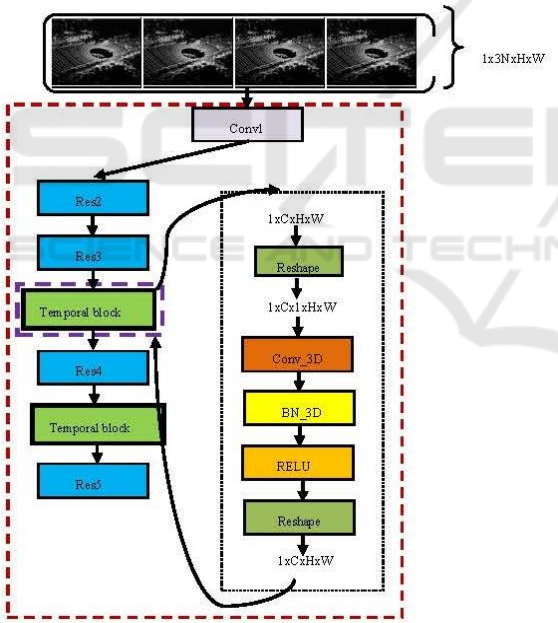


Figure 8: Spatial-temporal context network (STCN).

In order to jointly model the local spatial-temporal relationship, we leverage 2D convolution (whose input channel size is 3N) on each of the super-images. Specifically, the local spatial-temporal correlation is modelled by 2D convolutional kernels inside the Conv1, Res2, and Res3 blocks of SE-ResNext-50 as shown in Fig. (8). In our current setting, N is set to 4.

Temporal Block: 2D convolution on the super-images generates local spatial-temporal feature maps. Building the global spatial-temporal representation of the super-images is essential for understanding the context from successive point clouds. Specifically, we choose to insert two temporal blocks after the Residual 3rd and Residual 4th blocks of SE-ResNext (J. Hu et al., 2017) module. The temporal modelling blocks are designed to capture the long-range temporal dynamics inside a sequence of point clouds and they can be easily implemented by incorporating the architecture Conv3d-BN3d-ReLU. Applying two temporal convolutions on the local spatial-temporal feature maps after residual $3^{rd}$ (Res3) and $4^{th}$ blocks (Res4) introduces very limited extra computation cost but is effective to capture global spatial-temporal correlation progressively. In the temporal modelling blocks, weights of Conv3d layers are initially set to $1/(3xC_i)$, where $C_i$ denotes input channel size, and biases are set to 0. BN3d is initialized to be an identity mapping. In the training phase, we don't initialize the weights of SE-ResNext 2D convolution model from the ImageNet pre-trained model.

## 3 EXPERIMENTS

Here we conduct two types of experiments here. We compare our spatial-temporal multi-frame object detection with state-of-the-art 3D object detectors on new large-scale argoverse (M. Chang et al., 2019) dataset for autonomous driving. Second, we conduct experiments on two aspects: network architecture timing analysis and learnable parameter count.

### 3.1 Multi-frame Object Detection on Argoverse Dataset

#### 3.1.1 Implementation Details

We set the region of interest for the point cloud to $[-51.2,51.2] * [-51.2,51.2]$ meters and do BEV projection with a discretization resolution of 0.1 meter. We set the height range to $[-2.5, 1]$ meters in LIDAR. As a result, our input representation has the dimension of 968*968*3. We use data augmentation of rotation between $[-20, 20]$ degrees along the Z axis, global scaling along X, Y and Z dimensions with range to [0.95, 1.5] along with random flip along X axis during training. Network was trained from scratch without using any pre-trained model weights. The detection network is trained using Adam optimizer (Diederik P. Kingma et al., 2015) a learning rate of 1e-4 and a weight decay of 1e-4 for 300 epochs with a batch size of 4 on single RTX2080Ti GPU.

### 3.1.2 Evaluation Results

We compare mean average precision (mAP) at different IoU levels as a measure of accuracy. To the best of our knowledge, (El Sallab., 2018) is the only previous work which performs detection using temporal LiDAR point clouds. As shown in Table 1, our NLCN and STCN model accuracy outperforms both single- frame and CLSTM based 3D object detector (El Sallab., 2018) at all the IoU levels (0.5, 0.6, and 0.7).

Table 1: Ablation study of network performance on IoU thresholds.

| Model | AP@0.5IOU | AP@0.6IOU | AP@0.7IOU |
|---|---|---|---|
| Single-frame | .752 | .723 | .595 |
| Joint model | .759 | .716 | .561 |
| CLSTM | .786 | .74 | .579 |
| NLCN | .796 | .747 | .633 |
| STCN | .821 | .758 | .641 |

Also we show detailed timing analysis, FLOPS and learnable parameter count of each network architectures in Table 2. The computation of input representation and final NMS are both processed on CPU in Python. The network time is measured on a RTX-2080Ti GPU averaged over 100 sequential frames.

Table 2: Ablation study of network timing analysis and learnable parameter count.

| Model | Learnable params | FLOPS (in GMAC) | Speed (in ms) |
|---|---|---|---|
| Single-frame | 8.67M | 14.69 | 16 |
| Joint model | 8.68M | 62.46 | 74 |
| CLSTM | 9.89M | 24.67 | 27 |
| NLCN | 15.15M | 31.86 | 59.5 |
| STCN | 12.13M | 27.74 | 36 |

From the Table (1) and (2), we get the following observations: (1) NLCN model outperforms by 4.4mAP over single-frame 3D object detector and by 1.1mAP over YOLO4D 3D object detector. (2) STCN model outperforms by 6.9mAP over single-frame 3D object detector and by 3.5mAP over YOLO4D 3D object detector. (3) STCN outperform NLCN model by 2.5mAP @0.5IoU with less number of learnable parameters at a speed of 36ms.

# 4 CONCLUSIONS

In this paper, we introduce NLCN and STCN temporal context models that is able to tackle the tasks of BEV based object detection in the context of self-driving cars. This models not only leverages spatial information from current point cloud but also extract temporal feature from successive point clouds thereby enriching context information for the detection of 3D objects on LiDAR point clouds. By exploiting temporal information, our both the models NLCN and STCN outperforms single-frame object detection and CLSTM object detector by a large-margin. In the future, we plan to exploit HD maps and do End-to-End learning system of perception module in autonomous driving system.

# REFERENCES

J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Z. Kolter, D. Langer, O. Pink, V. Pratt, et al. 2011. Towards fully autonomous driving: Systems and algorithms. In *IEEE Intelligent Vehicles Symposium (IV)*

X. Chen, K. Kundu, Z. Zhang, H. Ma, S. Fidler, and R. Urtasun, 2016. Monocular 3d object detection for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

X. Chen, K. Kundu, Y. Zhu, H. Ma, S. Fidler, and R. Urtasun., 2017. 3d object proposals using stereo imagery for accurate object class detection. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*

2017. Baidu Apollo. http://apollo.auto...

2017. Google's Waymo Invests in LIDAR Technology, Cuts Costs by 90 Percent. https://arstechnica.com/cars/2017/01/googles-waymo-invests-in-lidartechnology-cuts-costs-by-90-percent/

2017. What it Was Like to Ride in GM's New Self- riving Cruise Car. https://www.recode.net/2017/11/29/16712572/general-motors-gm-new-selfdriving-autonomous-cruise-car-future.

2018. Volvo Finds the LIDAR it Needs to Build Self-Driving Cars. https://www.wired.com/story/volvo-self-driving-lidar-luminar/.

Redmon, J., Divvala, S., Girshick, R., Farhadi, 2016. A.: You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C., 2016. Ssd: Single shot multibox detector. In *European conference on computer vision, Springer*

M. Engelcke, D. Rao, D. Z. Wang, C. H. Tong, and I. Posner, 2017. Vote3deep: Fast object detection in 3d point clouds using efficient convolutional neural networks. In *International conference on Robotics and Automation (ICRA)*

B. Li, T. Zhang, and T. Xia., 2016. Vehicle detection from 3d lidar using fully convolutional network. In *Robotics: Science and Systems.*

Yang, B., Luo, W., Urtasun, R. 2018. Pixor: Real-time 3d object detection from point clouds. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*

M. Simon, S. Milz, K. Amende, and H.-M. Gross, Mar 2018. Complex-YOLO: Real-time 3D Object Detection on Point Clouds. In *European Conference on Computer Vision*

Ali, W., Abdelkarim, S., Zidan, M., Zahran, M., and El Sallab, A 2018. Yolo3d: End-to-end real-time 3d oriented object bounding box detection from lidar point cloud. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Luo, W., Yang, B., Urtasun, R, 2018. Fast and furious: Real time end-to-end 3d detection, tracking and motion forecasting with a single convolutional net. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*

S. Casas, W. Luo, and R. Urtasun, 2018. IntentNet: Learning to predict intention from raw sensor data, In *Proceedings. 2nd Annu. Conf. Robot Learning*

Wenyuan Zeng, Wenjie Luo, Simon Suo, Abbas Sadat, Bin Yang, Sergio Casas, and Raquel Urtasun, 2019. End-to-end interpretable neural motion planner. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*

El Sallab, A., Sobh, I., Zidan, M., Zahran, M., and Abdelkarim, S. Yolo4d, 2018: A spatio-temporal approach for real-time multi-object detection and classification from lidar point clouds. In *Thirty-second Conference on Neural Information Processing Systems, Workshop on Machine Learning for Intelligent Transportation Systems.*

M. Chang, J. Lambert, P. Sangkloy, J. Singh, S. Bak, A. Hartnett, D. Wang, P. Carr, S. Lucey, D. Ramanan, and J. Hays, 2019. "Argoverse: 3d tracking and forecasting with rich maps," In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, 2017. Feature pyramid networks for object detection. In *IEEE Computer Vision and Pattern Recognition (CVPR)*

Xiaolong Wang, Ross B. Girshick, Abhinav Gupta, and Kaiming He, 2018. Non-local neural networks. In *IEEE Computer Vision and Pattern Recognition (CVPR)*

S. Xingjian, Z. Chen, H. Wang, D. Yeung, W. Wong, and W. Woo, 2015. Convolutional LSTM network: A machine learning approach for precipitation now casting. In *Neural Information Processing systems(NIPS)*.

J. Hu, L. Shen, and G. Sun, 2017. Squeeze-and-excitation networks. *arXiv preprint arXiv:1709.01507*

Diederik P. Kingma and Jimmy Ba. Adam, 2015. A Method for Stochastic Optimization. In *International Conference on Learning Representations(ICLR)*