BLPNet: An End-to-End Model Towards Voxelization Free 3D Object Detection

Zhihao Cui
University of Technology of Sydney
Faculty of Engineering and IT
Sydney, Australia
zhihao.cui@student.uts.edu.au

Zhenhua Zhang
University of Technology of Sydney
Faculty of Engineering and IT
Sydney, Australia
zhenhua.zhang@student.uts.edu.au

Abstract—**Recent achievements in robotics application and autonomous navigation are accomplished through the fast progress in point cloud 3D object detection models. Many of the current single staged 3D object detection models rely heavily on the voxelization and PointNet based method for feature extraction. Although it provides an efficient way to process point cloud data, its lack of spatial and geometric relationships on both voxel and point level limited the detection accuracy of models. In this paper, we focus on these limitations of the voxelization based object detection pipeline and proposed a single-stage non-voxelization 3D object detection framework. This framework utilizes the bilateral convolution layer, region-based feature clustering and lattice to feature map layer to address the lack of spatial, and geometric relationships in both voxel and point level, which further improves 3D object detection accuracy on the KITTI dataset. Our method achieved 75.63 mAP in moderate difficulty and outperformed many influential object detection models on the KITTI benchmark leaderboard.**

Contribution—**We proposed a voxelization free single-stage model to achieve a desirable performance on the 3D object detection task.**

Keywords—3D object detection; robotics vision; lidar point cloud;

I. INTRODUCTION

3D object detection provides a unique advantage comparing to 2D image-based detections. Lidar sensor used in 3D object detection collects a set of 3D point coordinates, forming a point cloud data. This point cloud data is capable of representing the real-world environment and provides accurate 3D spatial information. In contrast, the 2D image suffers from a lack of accurate depth perception and real-world localization. These advantages of 3D object detection have applied to various applications such as autonomous driving [1, 2] house service robots [3] and object tracking for augmented reality [4].

The current 3D object detection network can be categorized into two general pipelines, single-stage, and two-stage model. In the 3D object detection task, the single-stage model commonly uses the voxelization method to evenly partition the unordered 3D point cloud space to an ordered grid space, subsequently with the predict branches for bounding box classification and regression. As the 3D point cloud possess spatial unordered property, it is necessary to voxelization the 3D space to applying the 3D convolution neural network to generate a feature map consequently to place anchor boxes upon to each feature map. For single-stage model VoxelNet [5], it voxelized the 3D space and followed by a Voxel Feature Extraction (VFE) to extract voxel-level features in each voxel. Next, the ordered voxels pass through a 3D convolution to generate a 3D feature map. Similar models like Second [6], Pointpillars [7] follow with the same voxelization procedure.

For the two-stage model, the first stage is usually used to coarse generate proposals, and the second stage is used to refine these proposals to improve the final prediction accuracy. Since the two-stage model does not need to give the final prediction result in the first stage, the proposal bounding boxes can be generated directly from each point without voxelized the 3D point cloud space. PointRCNN [8] applied PointNet++ to extract point-wise feature in an unordered 3D point cloud space, for each point it predicts the bounding box proposals with the high recall. F-PointNet [9] obtains the proposal from the off-the-shelf 2D object detection model. MV3D [10], AVOD [11] combine the 2D image and 3D points cloud projection view to generate the proposals.

In summary, we re-categorize the single-stage model as the voxelization model mostly due to it needs a voxelization procedure to construct an ordered 3D space consequently to achieve the single-stage 3D object detection pipeline. Conversely, the two-stage model is not necessary to voxelized the 3D space thus we call it a non-voxelization two-stage model. From the study, we found that the voxelized 3D space and deploy PointNet-like method to each voxel for feature extraction methods are highly used in many single-stage models. Although providing an efficient point cloud processing method, the voxelization based method possesses its problems. Voxels are extracted from evenly partitioned point cloud space. Within each voxel, feature extraction algorithms merge multiple down-sampled points to a fixed-length vector. The most common feature extraction algorithms used are PointNet or taking the average of point coordinates within each voxel. Taking average discard many of the information exists within each voxel, commonly results in loss of spatial and geometric information. For PointNet, due to its characteristics, the extracted features from multilayer perceptron and max-pooling do not establish the relationship between points but selecting a max feature from the individual point. This further influences the weak feature connection between voxels as well as between points. Furthermore, features extracted from voxels are isolated, provide no relationship context between voxels. Voxelization often requires additional hyperparameters such as the size of each voxel and the number of points per voxel. These hyperparameters directly influence the final feature map and the quality of features within each voxel. Tuning these parameters add complexities in obtaining optimal model overall. While the non-voxelization method does not possess the characteristic of voxelization. Its two-stage pipeline adds the additional network to the model, increasing its complexity. This results in slower inference and training time, which may be difficult to apply in real-time applications.

Our work is motivated by the single-stage method for its fast inference speed and its viability for real-time applications, such as autonomous driving and robot navigation. Model such as Pointpillars provides an optimized single-stage model with fast inference speed. As the problem stated above, we argue that the main bottleneck in single stage-based model is due to the lack of spatial and geometric relationships from features extracted on voxels. Base on this, we are aiming by improving the feature extraction method, the performance will improve while maintaining fast inference and training speed.

In this paper, we present a single-stage non-voxelization based 3D object detection model to address the issues stated above named BLPNet. The model composes of three core modules, Bilateral Convolution Layer (BCL) [12], region-based feature clustering and lattice to feature map layer. Point cloud spatial and geometrical information extracted from these three modules improved our model's performance. BCL directly convolves points to extract spatial and geometric features without voxelizing the point cloud space. The region-based feature clustering method processes and refines features extracted from the BCL. Subsequently, with the lattice to feature map layer, the refined features are projected onto a 2D feature map. We apply pattern recognition backbone such as region proposal network (RPN) [13] or feature pyramid network (FPN) [14] to the 2D feature map for final classification and regression.

The use of non-voxelized methodology distinguish our model from current voxel-based single staged object detection model. From above, our contribution could be summarized into three folds:

- We propose a non-voxelization single-stage 3D object detection framework that targets the autonomous driving scene. The framework abandoned the traditional voxelization method, furthermore obtained a voxel-free scheme which enhances the model performance while maintaining the efficiency on point cloud processing.
- We explore and demonstrate the feasibility of utilizing BCL beyond the segmentation task, and specially designed feature learning network applied in the object detection task.
- We also propose region-based feature clustering and lattice to feature map layer which refines and bridge point features to pattern recognition backbone.

II. Related Work

A. *Voxelization based single-stage pipeline*

Voxelization based models [5, 6, 7] evenly partition 3D point cloud space into a regular voxel grid; Voxel-level features are extracted by using a method similar to PointNet [15]. These features are arranged together to form a new feature map that is processed subsequently by 3D sparse convolution [6], or project to a 2D plane to proceed 2D convolutions [7]. These models employ the voxelization method to convert the unordered point cloud to an ordered voxel. Voxelization method requires downsampling to maintain a constant number of points in voxels. Within each voxel, PointNet/averages extraction method is applied to extract features. This extraction method results in information loss and lacking in the integrity of point relationships. Our proposed model aims to tackle these problems in the current single-stage method through the proposed non-voxelization pipeline. In other words, we directly process the point cloud data without voxelization to preserve the spatial connectivity between each point.

B. *Non-voxelization based two-stage pipeline*

MV3D [10], AVOD [11], MMF [16] exploits multisensory information in addition to the point cloud data. MV3D generates proposals from point cloud bird's eye view (BEV) perspective. Features are extracted from proposals projected in a 2D image, front view, and bird's eye view perspectives. These features are fused for second stage refinement. AVOD refined the proposal generation process of MV3D by fusing 2D and BEV features before obtaining proposals. MMF exploits sensory information further by fusing online map, ground-relative BEV representation, depth image, RGB image, and pseudo LIDAR to generate better proposals. These two-stage methods do not employ voxelization for its point cloud processing but require complex model architectures to achieve high detection accuracy. With the increase in model complexity, the training and inference time hence increases. To this end, we employ the non-voxelization method in single-stage models to further improve the detection accuracy.

C. *Methods directly process point cloud*

PointNet [15] uses a combination of multilayer perceptron and max-pooling to directly extract features from the unordered point cloud. However, the feature extracted from PointNet lacks local and geometric information. Base on PointNet, PointNet++ [17] only addresses the lack of local information by extracting features from the clustered point cloud. Splatnet [18] addresses PointNet's both issues by using BCL. BCL maintains spatial and geometric features between points. Our model adopts the BCL as our feature learning network. Using this layer allows direct convolution on point cloud without the need for voxelization, obtaining finer features to enhance the learning of the whole model.

III. Our Model

We present our Non-voxelization single-stage model BLPNet for 3D bounding box estimation. Figure 1 demonstrates the general pipeline of our model, which

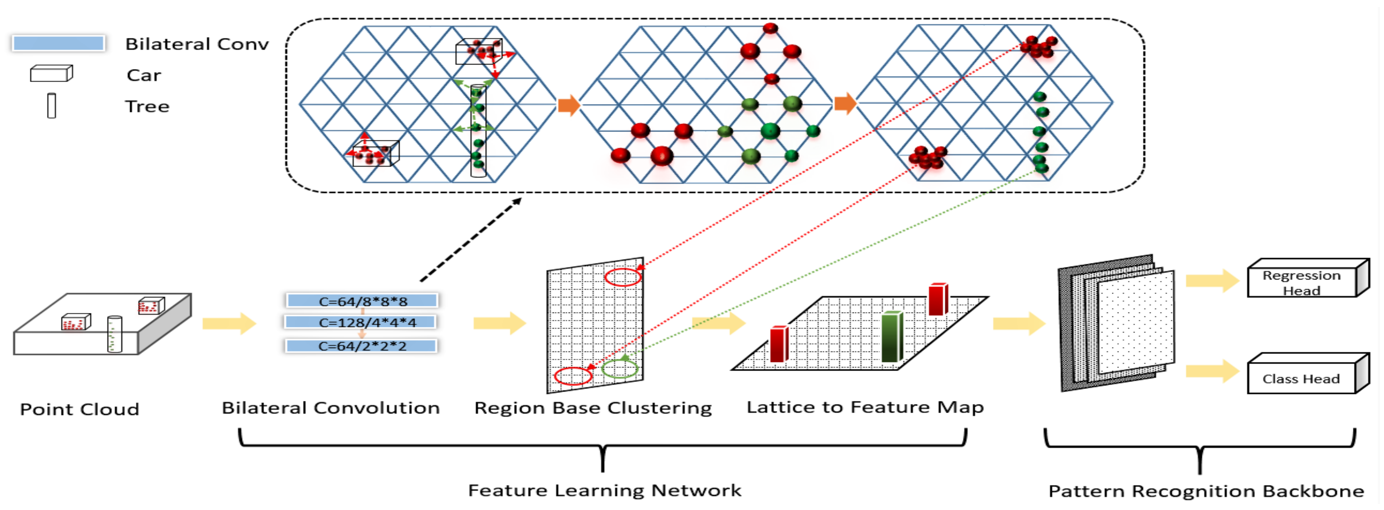

Figure 1 Overall pipeline of BLPNet. The feature learning network consists of the BCL feature extraction layer, Region-based feature clustering layer and point to feature map layer. We use region proposal network as pattern recognition backbone in the model. The backbone can be replaced with other achitectures.

consists of two major parts feature learning network and pattern recognition backbone.

Our feature learning network is formed by three components, BCL, region-based feature clustering and lattice to feature map layer. The BCL convolves the input point cloud directly without voxelization, generating higher dimension features. In region-based feature clustering, we aggregate all the generated features within each feature map region without limiting the number of points, unlike voxelization methods [5, 6, 7]. The features are further refined through a multilayer perceptron layer, which provides linear transformation operation and dimension reduction. The lattice to feature map layer projects the output of the region-based feature clustering to a 2D feature map for efficient 2D convolution.

In our model, we apply the region proposal network like [5] and [7] as our pattern recognition backbone. This backbone can be replaced with other architectures like feature pyramid network (FPN) [14].

A. *Bilateral Convolution Layer*

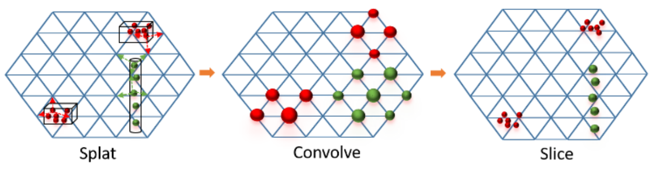

Figure 2 Visual demonstration of the splatting, convolving and slicing steps in the bilateral convolution layer. The blue lattice hyperplane is consisted of uniformed simplex.

In this section, we will briefly introduce the Bilateral Convolution Layer (BCL) [12], which is derived from the permutohedral lattice Gaussian filtering proposed by Andrew Adam [19]. BCL is a sparse high-dimensional convolution method with three main stages: splat, convolve, and slice. BCL possesses the capability of directly convolve on unordered point cloud data by projecting the point feature into a lattice hyperplane during the splat stage. The projected point feature in the hyperplane is merged into the closest vertex within a simplex using barycentric interpolation. Slice projects the convolved vertices back to original space based on weights calculated from the interpolation. Figure 2 provides a visual representation of the three steps which form the BCL.

BCL takes two inputs, lattice feature and points feature. Base on the lattice feature, the point feature is projected onto the lattice hyperplane constructed from permutohedral lattice [19]. We use the Cartesian coordinate of the point cloud as lattice feature for point feature projection. The receptive field of the BCL can be controlled by scaling the lattice feature [18]. Scaling up the lattice feature decreases the number of features projected into the same simplex, resulting in fewer features merging into the same vertex in the splat phase. This decreases the receptive field as fewer features are convolved together. Decreases the scale increases the number of features project onto the same simplex, causing more features to merge into the same vertex. This increases the receptive field as more features are convolved together.

B. *Lattice to Feature Map Layer*

Lattice to feature map layer projects the clustered point feature to 2D feature map with transformed feature map indices. We initially define the 2D feature map to be shape H and W. Before projection, BCL extracts point cloud feature C_{xyzr} to higher dimension point feature C_n . The corresponding point coordinates xyz are transformed into the feature map index H_i, W_i by dividing the xyz coordinates by the feature map grid size in point cloud space. The size of the feature map grid can be determined by dividing the range of point cloud space by the shape of the 2D feature map. We discard the z-axis' index transformation to maintain the 2D feature map shape. Region-based feature clustering takes the extracted point feature C_n and transformed index H_i, W_i, outputs aggregated feature C' and unique index H_i, W_i. Lattice to feature map layer projects the aggregated feature C' base on the unique indices H_i, W_i into the generated 2D feature map. Figure 3 demonstrate the overall process of

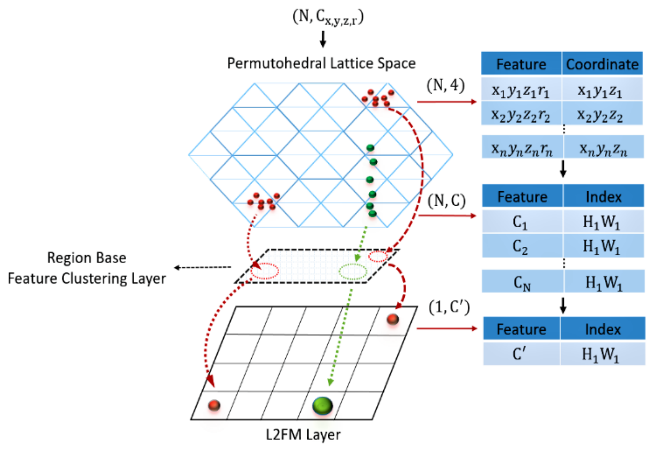

Figure 3 Points feature extracted from BCL are clustered by region-based feature clustering. Through lattice to feature map layer, the clustered features are projected onto a 2D feature map. Tables at the right demonstrate the overall process. $\mathbf{C_N}$ indicates the N-th feature in a cluster. $\mathbf{H_1}, \mathbf{W_1}$ indicates the first feature map index.

clustering and projection, the data structure of a single cluster during this process is shown in the table on the right.

C. *Region-based feature clustering*

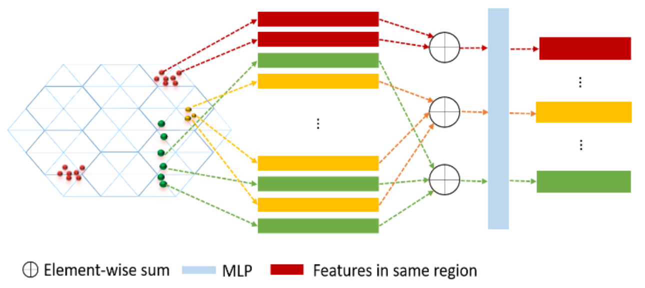

Figure 4 Visual representation of region-based feature clustering. Rectangles with the same color indicating features in the same region. The element-wise sum is applied to points features with the same color and followed by a multi layered perceptron layer.

We noted that from [5, 6, 7], they required to sample the limited number of points within each voxel before extracting features. This method neglects the number of points and breaks the integrity of the point cloud. We overcome this challenge by region-based feature clustering to aggregate all the point features within a region. This method is capable of process uneven number of features in each feature map region and fuses into a single feature.

As demonstrated in Figure 4, the point features extracted from the BCL are unordered and scattered. Although unordered, each feature possesses its xyz coordinate in the point cloud space. Based on the coordinate, we can obtain the feature map index H_i, W_i for each feature. Through regional clustering, the features with the same feature map index are grouped. We applied the element-wise sum between these clustered features, produce a fixed-length feature containing spatial information for each cluster. Furthermore, we use a single multi-layered perceptron to refine the clustered features before projection occurs inside the lattice to feature map layer. Our model benefited from this method which aggregates all the features without limiting the number of features per feature map. The experiment details are shown in *Ablation study section C.*

D. *Loss Function*

For the binary classification problem, we use focal loss [20] to handle the imbalanced data problem. During training, only positive and negative anchors are used to compute loss, the rest of the anchors are neglected and not considered in loss computation. In General, binary classification focal loss is the sum of all positive and negative anchor loss:

$$\mathrm{L_{cls}} = \sum_{p\,\in\,pos,neg} -\alpha(1-\mathrm{p^a})^{\gamma}\, log\, \mathrm{p^a} \tag{1}$$

The default setting of alpha (α) = 0.25 and gamma (γ) = 2 to train our model. The probability (p^a) is the prediction score of the anchors.

We use SmoothL1Loss [21] to compute regression loss for each bounding box. 3D box regression is defined as $(\Delta x, \Delta y, \Delta z, \Delta h, \Delta w, \Delta l, \Delta\theta)$. Δ denotes the offset between anchors box and ground truth box. SmoothL1Loss computes the loss between predicted Δ_{pred} and ground truth Δ_{gt}.

$$\Delta = (\Delta_{\mathrm{pred}} - \Delta_{\mathrm{gt}}) \tag{2}$$

$$\mathrm{smooth_{L1}}(\Delta)_{\Delta\in(\mathrm{x,y,z,h,w,l,\theta})} = \begin{cases} 0.5(|\Delta|)^2, & if\ |\Delta| < \frac{1}{\delta^2} \\ |\Delta| - 0.5, & otherwise \end{cases} \tag{3}$$

We define sigma (δ) to be 3 in our implementation. The total localization loss is defined as the sum of individual positive anchor's SmoothL1 loss.

$$L_{loc} = \sum_{\Delta\in(\mathrm{x,y,z,h,w,l,\theta})} smooth_{L1}(\Delta) \tag{4}$$

The total loss is the sum of Focal Loss and SmoothL1Loss. N_{pos} is the number of positive anchors. The total loss is as follows:

$$L = \frac{1}{N_{pos}}(L_{cls} + L_{loc}) \tag{5}$$

IV. Implementation Details

A. *Data preparation*

We observed during training, randomly sample points reduce the prediction accuracy of our model. We were able to conclude that due to the sparsity nature of the point cloud collected from the Lidar sensor, points closer with denser points are more likely to be sampled. Hence in our training, we calculate the mean distance value of all points in a point cloud and sample all of the points with a distance greater than the mean distance, then sample the remaining points less than the mean distance, which in total sampled 16,000 points.

We used the anchor box matching method similar to [7]. The anchor boxes are matched to a ground truth box base on the positive and negative matching threshold. The threshold is defined base on the 2D intersection over union (IoU) overlap between anchor box and ground truth box. If the IoU is above the positive threshold, then the anchor box is assigned to the matched box class and is considered as a positive anchor. If IoU is below the negative threshold, the anchor box is assigned as the background class and is considered a negative anchor.

For the car class, an anchor box is positively matched if the IoU between the anchor box and corresponding ground truth box is greater than 0.6. An anchor box is negatively matched if the IoU is below 0.55. Anchor boxes with IoU in between positive and negative thresholds are neglected. Before matching, we generate anchor boxes base on the class. For Car class, we create the anchor box with width, length and height of (1.6m, 3.9m, 1.5m). All anchor boxes are generated with two orientations, 0 and 90 degrees. The point cloud space is constrained within a range of (x_{min}, y_{min}, z_{min}, x_{max}, y_{max}, z_{max}) for Car class is at (0, -40, -3, 70.8, 40, 1).

B. Feature Engineering

We employed additional point-wise handcraft feature to further improve the detection accuracy of our model. Before extracting features from BCL, we compute the Euclidean distance between each lattice to the feature map region center coordinate. This computed distance is concatenated to its corresponding point coordinate as final feature, which is further processed by the BCL layers.

C. Feature Learning Network

We employed three BCL layers in feature learning network to extract higher dimension point features. First, we set the lattice feature to the point cloud coordinate xyz. The input feature of the first layer BCL is the point cloud coordinate and Lidar reflection intensity. Then, to establish a hierarchical network structure, we individually multiply scale factor of 8, 4 and 2 to each of the three BCL layer's lattice feature to achieve small to high receptive field. The details of BCL layer architecture experiments are shown *in Ablation study section B*

The features extracted through BCL are clustered by region-based feature clustering. During clustering, we compute the feature map index of the point feature. The index is obtained by dividing the point feature's corresponding point cloud coordinate by the grid size of the feature map. The grid size can be computed from dividing the feature map shape (368, 320) by the real point cloud range (70.8, 80, 4) meters. Note only x (70.8) and y (80) is used to compute the index. The region-based feature clustering groups point features with the same feature map index, and outputs clustered feature with a unique index. Based on the unique index, the clustered features are projected onto the feature map.

D. Pattern Recognition Backbone

We utilized region proposal network as pattern recognition backbone in our implementation. The region proposal network composed of three scales feature maps with bottom-up pyramid architecture. Each scale is processed through several convolution layers while maintaining its feature map size. At the end of several convolutions, all three features maps are upsampled to the same size via transposed convolution layer. The three upsampled layers are then concatenated together for bounding box regression and classification.

In our implementation, the three scales feature maps are sizes of (184, 160), (92, 80) and (46, 40). The initial feature map scale (184, 160) is downsampled from the projected feature map of shape (368, 320) via 3×3 2D convolution with stride 2 and padding 1. The above feature map scales are convolved 3, 5 and 5 times with filter size of 64, 128, and 256 respectively. The convolution kernel is 3×3 with stride 1 and padding 0. Then, the three scale feature maps' output is upsampled to size (184, 160) with filters of 128 before concatenating together. The concatenated feature map will follow by two 1×1 2D convolution layers for classification and regression. All the layer output except classification and regression convolution layers are followed by BatchNorm [22] and ReLu [23].

V. EXPERIMENT SETUP

A. Dateset

We evaluate BLPNet base on the KITTI 3D object detection benchmark [1]. The dataset consists of images and point clouds with 7,481 training and 7,518 testing data splits. The evaluation of the dataset is based on three categories: Car, Pedestrian and Cyclist. Each class is further evaluated on three difficulties: easy, moderate and hard. The difficulties of each class are determined based on object occlusion and the level of truncation. Average precision (AP) is used to evaluate the prediction results for the three classes on three difficulties. We split the training data set into 3,712 training samples and 3,769 validation samples following [6]. We train on only Lidar point cloud and compare against models that utilize single-stage voxelization or two-stage non-voxelization methods.

B. The training schemes

The network is trained on Nvidia Quadro P5000 graphics card for 20 epochs with batch size 8 and weight decay 0.999. We used the modified one cycle learning rate [24] with a minimum learning rate of 0.0001 and maximum learning rate of 0.001. Also, we employed Adam optimizer [25] with minimum β_1 0.85, maximum β_1 0.95 and β_2 0.9999.

During the training phase, we conduct data augmentation of a random flip, scaling with a scale factor sampled from [0.95, 1.05] and rotation around the vertical Y-axis between [-pi/4, pi/4]. Due to the relatively small number of positive samples in some scenes, we employed the ground truth database sampling technique inspired by Second [6] to continuously sample ground truth into current point cloud training data. The sampling technique enriches the positive samples, reduces the amount of label imbalance during training.

At the evaluation stage, we used the same sampling strategy in training and sample 12000 points to reduce the inference time. The detection result is selected from rotated non-maximum suppression with the confidence threshold of 0.3 and IOU threshold of 0.05.

VI. RESULT

A. 3D Object Detection on KITTI

We evaluate BLPNet's performance base on the KITTI dataset [3]. The KITTI dataset has three evaluation metrics: 2D, 3D and bird's eye view (BEV). These three metrics are

Table I BLPNet compares against other state of the art models on KITTI test set. ***V2*** = Voxelization Two-stage; ***NV2*** = Non-Voxelization Two-stage; ***V1*** = Voxelization Single-stage; ***NV1*** = None-Voxelization Single-stage

Model	Method Base	Car AP (IoU=0.7)			Pedestrians AP (IoU=0.5)			Cyclists AP (IoU=0.5)		
		Moderate	Easy	Hard	Moderate	Easy	Hard	Moderate	Easy	Hard
MV3D [10]	v2/ lidar+rgb	62.35	71.09	55.12	N/A	N/A	N/A	N/A	N/A	N/A
F-PointNet [9]	v2/ lidar+rgb	70.29	81.20	62.19	**44.89**	51.21	40.23	56.77	71.96	50.39
ContFuse [26]	v2/ lidar+rgb	68.78	83.68	61.67	N/A	N/A	N/A	N/A	N/A	N/A
AVOD-FPN [11]	v2/ lidar+rgb	71.88	81.94	66.38	42.81	50.80	40.88	52.18	64.00	46.61
PointRCNN [8]	nv2/ lidar	**76.51**	85.94	68.32	41.78	49.43	38.63	59.60	73.93	53.59
VoxelNet [5]	v1/ lidar	65.11	77.47	55.12	33.69	39.48	31.51	48.36	61.22	44.37
Pointpillars [7]	v1/ lidar	74.99	79.05	68.30	43.53	**52.08**	41.49	59.07	**75.78**	52.92
Second [6]	v1/ lidar	73.66	83.13	66.20	42.56	51.07	37.29	53.85	70.51	46.90
BCLNet (ours)	nv1/ lidar	75.63	**86.80**	**68.88**	43.59	51.27	**42.49**	**60.30**	75.62	**53.93**

Table II Model performances on different methods used measured in Car class at IoU = 0.7 on KITTI val set

Pointwise feature	Feature extraction method	Clustering method	Multilayer perceptron	AP (IoU=0.7)		
				Easy	Moderate	Hard
×	BCL	Sample&Concat	×	85.58	73.91	64.46
×	BCL	Region Based	×	86.20	76.63	73.36
×	BCL	Region Based	✓	87.52	77.62	74.57
✓	PointNet	Region Based	✓	86.79	76.27	67.35
✓	BCL	Region Based	✓	**89.07**	**78.49**	**76.06**

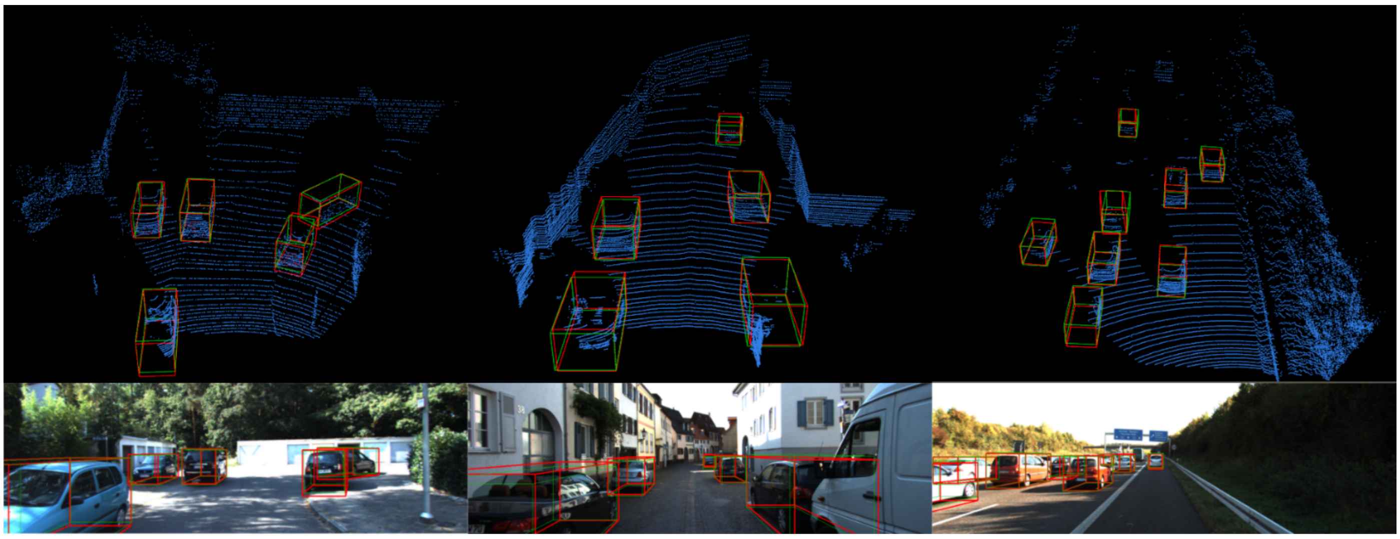
Figure 5 Qualitative results of BLPNet on KITTI dataset. The detections are under different road environments with ***red*** bounding box indicates prediction and ***green*** indicates ground truth box.

further evaluated under different IoU and difficulties under IoU at 0.5 and 0.7 with easy, moderate and hard difficulties. For our model, we will evaluate the results with the 3D metric under IoU=0.7. Table I shows the comparison of our model against the other state of the arts 3D object models on the test set. As demonstrated in Table I, our model outperform models not only with Lidar only method but also multiple fusion-based methods at moderate difficulty. Comparing closely, our Lidar based non-voxelization method outperformed AVOD-FPN, ContFuse and F-PointNet multi-sensor fused method by a large margin. In Lidar only methods, our non-voxelization model was able to outperform voxelization method Voxelnet, SECOND and Pointpillars especially in car detection.

B. *Qualitative results*

Figure 5 provides a visual representation of the 3D box prediction in both point cloud and image for BLPNet. Under different road condition, our model predicts accurate bounding box dimension and location, overlapping closely to the ground truth box.

VII. ABLATION STUDY

In this section, all experiments are trained on the KITTI training split dataset and evaluated on the validation split dataset with the *car* class at IoU = 0.7 mAP metric.

Table III Performance of different lattice scale and feature dimension on KITTI validation split in easy, moderate and hard average precision at IoU above 0.7

Lattice Scale	BCL layer depth	AP (IoU=0.7)		
		Easy	Moderate	Hard
[8,4,2,1]	[64,128,128,64]	86.33	76.56	68.79
[4,2,1,0.5]	[64,128,128,64]	83.63	69.01	66.99
[8,4,2]	[64,128,64]	**89.07**	**78.49**	**76.06**
[4,2,1]	[64,128,64]	85.79	76.88	72.03
[2,4,8]	[64,128,64]	87.25	75.81	71.10
[8,4]	[64,64]	86.08	76.07	69.02
[4,2]	[64,64]	83.59	71.04	64.76

A. *Module performance*

We analyzed the effectiveness of our proposed modules as demonstrated in Table II. To validate the effect of feature extracted from BCL, we replace BCL layers with PointNet in the feature learning network. In the results, BCL outperformed PointNet by 2.3% under moderate, 2.2% under easy and 8.7% under hard. The improved performance between PointNet and BCL showed the importance of the spatial and geometric features extracted by BCL especially on hard mode. These features contain more descriptors on the objects within the point cloud, hence improve the detection of objects.

To evaluate the influences of region-based feature clustering, we compare it to another clustering method. For point features extracted from BCL, we sample fix number of features at each feature map index and concatenate the number of features to its channel. Region-based method increased by (easy=0.6, moderate=2.7, hard=8.9) % compared to the sample and concatenate method from above. This result demonstrates the improvement in using all point features instead of sampling fixed-length features. In addition, adding one layer of multilayer perceptron further improved the performance by (easy=1.3, moderate=1, hard=1.2) %.

The handcraft pointwise feature improved our model by (easy =1.6, moderate = 0.9, hard = 1.5) %. We further tested different handcraft features including density distribution of points and PCA analysis, which all resulted in reduced in performance of our model.

B. *Feature extraction BCL Architecture*

To build a suitable feature extraction layer, we conducted the following experiments to explore the characteristics of BCL. As shown in Table III, we analyzed the BCL layers based on the following aspects: BCL network depth, lattice feature scaling and BCL skip layer.

In our implementation, three BCL layers performed the best out of all the tests. We further increased and decreased the BCL layer but resulted in lower performance of the model, especially under hard difficulty. Under the best scaling configuration, 2-layer depth decreased by (easy=3, moderate=2.4, hard=7.04) % comparing to 3-layer depth. 4 BCL layer depth slightly decreases the performance by (easy=2.7, moderate=1.9, hard=7.27) %. Similar to CNN, increasing BCL layer depth raises the number parameters in a model, allowing the model to further converge to the dataset distribution. But the inadequate selection of BCL layers results reduces in performance, as demonstrated in the above experiments.

Table IV Different method of fusion on region-based feature clustering

Region Base Cluster	AP (IoU=0.7)		
	Easy	Moderate	Hard
Sum	**88.07**	**77.49**	**75.06**
Mean	83.60	74.47	67.58
Max	85.92	75.24	67.92

Table V Model inference speed comparisons.

Model	Method Base	Speed (ms)
MV3D [10]	voxel Two-stage	360
F-PointNet [9]	voxel Two-stage	170
ContFuse [26]	voxel Two-stage	60
AVOD-FPN [11]	voxel Two-stage	100
PointRCNN [8]	Non-voxel Two-stage	100
VoxelNet [5]	voxel Single stage	500
Pointpillars [7]	voxel Single stage	16
Second [6]	voxel Single stage	38
BCLNet (ours)	**Non-voxel Single stage**	**49**

To test the effect of the receptive field of the BCL layer, we apply different lattice scaling to our model. From Table 3, scaling the lattice feature from 8 to 2 under three-layer depths performed the best (easy=88.07, moderate=77.49, hard=75.06) %. Downscaling the lattice feature from 4 to 1 significantly reduces the performance by (easy=3.3, moderate=1.6, hard=4.0) % comparing with the best performance.

We have also tested different layer fusion method on our BCL feature extraction architecture. Layer wise sum or concatenate were applied to the BCL layers in different sequences, the resulting performance has almost no influence on the prediction accuracy of the model. We believe that the BCL itself was capable of learning the spatial features of the point cloud to full potential. Further manipulation of layer fusions resulting in similar features that provide no further improvements on the performance of the model.

C. *Region based clustering fusion methods*

We explored different methods of fusion including element-wise sum, mean and max. The results are demonstrated in Table IV. From these tests, we discover that element-wise sum fusion method performs the best compared to mean and max. Furthermore, applying one layer of multi-layered perceptron after fusion further improves the detection accuracy.

D. *Model efficiency*

Table V compares our model inference time with the recently stage-of-arts mode. Our non-voxelization model inference speed outperformed all of the two-stage models and achieved similar speed as the single-stage voxelization model but with higher accuracy.

VIII. CONCLUSION

In this paper, we have proposed a non-voxelization based 3D object detection model BLPNet to address the lack of spatial and geometric features in current 3D object detection models. Our model employed the BCL, region-based feature clustering and lattice to feature map methods. With the characteristics of BCL and the proposed lattice to feature map layer, the model refines and project point cloud spatial features directly into a 2D feature map for efficient convolution operation. The model explored the lack of spatial and geometric features in current single staged models and provided a non-voxelization pipeline to address this issue. Our model demonstrated its improvement in performance and efficiency on the KITTI dataset against other proposed Lidar only and Lidar/image fusion 3D object detection models.

References

[1] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Vision meets robotics: The KITTI dataset, *The International Journal of Robotics Research,* vol. 32, *(11),* pp. 1231-1237, 2013.

[2] Ruben Gomez-Ojeda, Jesus Briales, and Javier Gonzalez-Jimenez, PL-SVO: Semi-direct monocular visual odometry by combining points and line segments, in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS),* 2016, pp. 4211-4216.

[3] Yong-Joo Oh and Y. Watanabe, Development of small robot for home floor cleaning, in *Proceedings of the 41st SICE Annual Conference. SICE 2002.* 2002, pp. 3222-3223.

[4] Youngmin Park, Vincent Lepetit, and Woontack Woo, Multiple 3d object tracking for augmented reality, in *Proceedings of the 7th IEEE/ACM International Symposium on Mixed and Augmented Reality,* 2008, pp. 117-120.

[5] Yin Zhou and Oncel Tuzel, Voxelnet: End-to-end learning for point cloud based 3d object detection, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* 2018, pp. 4490-4499.

[6] Yan Yan, Yuxing Mao and Bo Li, Second: Sparsely embedded convolutional detection, *Sensors,* vol. 18, *(10),* pp. 3337, 2018.

[7] Alex. H. Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. PointPillars: Fast encoders for object detection from point clouds, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* 2019, pp. 12697-12705.

[8] Shaoshuai Shi, Xiaogang Wang and Hongsheng Li, Pointrcnn: 3d object proposal generation and detection from point cloud, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* 2019, pp. 770-779.

[9] Charles Ruizhongtai Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J. Guibas. Frustum pointnets for 3d object detection from rgb-d data, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* 2018, pp. 918-927.

[10] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* 2017, pp. 1907-1915.

[11] Jason. Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven Lake 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.

[12] Varun Jampani, M artin Kiefel, and Peter. V. Gehler, Learning sparse high dimensional filters: Image filtering, dense crfs and bilateral neural networks, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* 2016, pp. 4452-4461.

[13] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks, in *Advances in Neural Information Processing Systems,* 2015, pp. 91-99.

[14] Tsung-Yi Lin, Piotr Dollar, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 2117-2125.

[15] Charles Ruizhongtai Qi, Hao Su, Kaichun Mo, and Leonidas J.Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* 2017, pp. 652-660.

[16] Ming Liang, Bin Yang, Yun Chen, Rui Hu, and Raquel Urtasun. Multi-task multi-sensor fusion for 3D object detection, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* 2019, pp. 7345-7353.

[17] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space, in *Advances in Neural Information Processing Systems,* 2017, pp. 5099-5108.

[18] Hang Su, Varun Jampani, Deqing Sun, Subhransu Maji, Evangelos Kalogerakis, Ming-Hsuan Yang, and Jan Kautz. Splatnet: Sparse lattice networks for point cloud processing, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* 2018, pp. 2530-2539.

[19] Andrew Adams, Jongmin Baek, and Myers. A. Davis, Fast high-dimensional filtering using the permutohedral lattice, in *Computer Graphics Forum,* 2010, pp. 753-762.

[20] Tsung. Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar, Focal loss for dense object detection, in Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 2980-2988.

[21] Ross Girshick, Fast r-cnn, in Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1440-1448.

[22] Sergey Ioffe and Christian Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, *ArXiv Preprint arXiv: 1502.03167,* 2015.

[23] Vinod Nair and Geoffrey. E. Hinton, Rectified linear units improve restricted boltzmann machines, in *Proceedings of the 27th International Conference on Machine Learning (ICML-10),* 2010, pp. 807-814.

[24] Leslie. N. Smith, Cyclical learning rates for training neural networks, in *2017 IEEE Winter Conference on Applications of Computer Vision (WACV),* 2017, pp. 464-472.

[25] Diederik. P. Kingma and Jimmy Ba, Adam: A method for stochastic optimization, *ArXiv Preprint arXiv: 1412.6980,* 2014.

[26] Ming Liang, Bin Yang, Shenlong Wang, and Raquel Urtasun, Deep continuous fusion for multi-sensor 3d object detection, in *Proceedings of the European Conference on Computer Vision (ECCV),* 2018, pp. 641-656.