IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 19, 2021, accepted March 29, 2021, date of publication April 1, 2021, date of current version April 9, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3070379

RoIFusion: 3D Object Detection From LiDAR and Vision

CAN CHEN, LUCA ZANOTTI FRAGONARA, AND ANTONIOS TSOURDOS
School of Aerospace, Transport and Manufacturing, Cranfield University, Cranfield MK43 0AL, U.K.
Corresponding author: Luca Zanotti Fragonara (l.zanottifragonara@cranfield.ac.uk)

ABSTRACT When localizing and detecting 3D objects for autonomous driving scenes, obtaining information from multiple sensors (e.g., camera, LIDAR) is capable of mutually offering useful complementary information to enhance the robustness of 3D detectors. In this paper, a deep neural network architecture, named RoIFusion, is proposed to efficiently fuse the multi-modality features for 3D object detection by leveraging the advantages of LIDAR and camera sensors. In order to achieve this task, instead of densely combining the point-wise feature of the point cloud with the related pixel features, our fusion method novelly aggregates a small set of 3D Region of Interests (RoIs) in the point clouds with the corresponding 2D RoIs in the images, which are beneficial for reducing the computation cost and avoiding the viewpoint misalignment during the feature aggregation from different sensors. Finally, Extensive experiments are performed on the KITTI 3D object detection challenging benchmark to show the effectiveness of our fusion method and demonstrate that our deep fusion approach achieves state-of-the-art performance.

INDEX TERMS Sensors fusion, 3D object detection, Region of Interests, neural network, segmentation network, point cloud, image.

I. INTRODUCTION

Object detection with 3D bounding boxes is one of the fundamental challenges of situational awareness and environmental perception of autonomous systems (e.g., autonomous vehicles, robots, unmanned aerial vehicles, etc.). In fact, autonomous systems need to perceive objects in their surrounding environment using different sensors (e.g., cameras, LIDAR) for navigation and obstacle avoidance. In the past few years, 2D object detection is one of the area of computer vision that made the most significant progress [1]–[12], especially with the advent of convolutional neural network (CNN) technology [13]. However, 3D object detection remains an open challenge, especially when multiple, heterogeneous sensors are used to obtain more diverse and robust information.

Recently, many researchers focused on the exploitation of point-cloud based methods for 3D object detection due to the advantages of this type of data provide: precise depth information and dense geometric shape features [14]–[19]. Surprisingly, recent approaches [20]–[25] outperform even fusion-based methods.

The associate editor coordinating the review of this manuscript and approving it for publication was Sudipta Roy.

However, a standard point cloud is incapable of offering high-resolution texture information of an object, which typically is very beneficial in capturing discriminative features. In contrast, images provide rich colour and texture information, but with a lack of depth and scale information without resorting to complex and computationally intensive algorithms (e.g., stereography). For example, objects (e.g., cars, pedestrians) detected at long-distance results in few points, which makes the classification or localization of these objects very difficult resorting to point clouds data only. Meanwhile, in the image domain, texture and colour features of objects can still be visible even at longer distances, due to the higher spatial density of images, and likely to be captured by existing mature 2D CNNs technology. As a result, the fused features, leveraging the advantages from both point clouds and images, are beneficial in exploiting more reliable representations and improving the performance of the 3D object detection architecture.

The development of an efficient and effective sensor fusion method proves yet challenging due to the viewpoint misalignment of point clouds and images. In order to address this issue, early methods *MV3D* [26] and *AVOD* [27] propose 3D bounding box regression on fused 2D images and 2D Bird-Eye-View (BEV) feature maps, although quantization for BEV generation gives rise to a lot of geometric

information losses. *Frustum Pointnets* [28] and *Pointfusion* [29] both project 2D bounding boxes from the image-based 2D detector onto the point clouds to coarsely cluster potential foreground points. Then, PointNets can be applied for 3D bounding box estimation, but the overall procedure heavily relies on the performance of 2D detectors. *PointPainting* [30] feeds the pixel-wise semantic features captured from the image-based semantic segmentation model onto corresponding point-wise semantic features in the point cloud to boost the performance of 3D object detection.

It can be observed that the main disadvantage of using dense point-pixel fusion methods such as [30] is that it leads to a considerable amount of redundant computations. Meanwhile, using a BEV-image fusion method allows the deep learning-based fusion of the feature maps captured from an individual viewpoint but with geometric information losses. However, the authors believe that it is not strictly necessary to densely fuse the whole point clouds with images. Conversely, it is feasible to generate a small set of potential Region of Interests (RoIs) in the point clouds and the images, followed by applying a deep fusion method only on those local regions used for 3D object detection. The advantages of this fusion method are that it considerably reduces the computation cost and allows an easy alignment of the viewpoints on the local regions.

Aiming at filling the aforementioned gap, we hereby present an efficient and lightweight deep fusion method for 3D object detection for point clouds and images. Our main contributions can be summarized as follows:

- We propose a lightweight deep fusion neural network, named *RoIFusion*, aiming at sparsely fusing a small set of RoIs from the point clouds and the images for 3D object detection, which is beneficial for avoiding dense point-pixel fusion.
- We propose a fused keypoints generation (FKG) layer to estimate a small set of keypoints on the objects for further ROIs generation, followed by a voting layer used to generate the center points of the objects.
- We propose a RoIFusion layer to generate and aggregate 2D/3D RoIs for feature fusion, followed by a prediction layer to regress the parameters of the 3D bounding boxes and their orientation.

II. RELATED WORK

A. LIDAR-ONLY METHODS FOR 3D OBJECT DETECTION

Many existing methods explored the possibility of detecting the objects with 3D bounding boxes only using point cloud data, as they provide accurate geometric information. It is possible to broadly classify these methods into four categories: projection-based methods, volumetric-based methods, pointnets-based methods, and point-voxel methods.

1) PROJECTION-BASED METHODS

Several works [23]–[25] apply 2D CNNs directly to Bird-Eye-View (BEV) projected from the raw point clouds in order to estimate the 3D bounding box and orientation of an object. *FVNet* [31] projects the raw point clouds to the front view, which is then fed to a proposal generation network and a refinement network to estimate the parameters of the 3D bounding box (i.e. object location, size, and orientation). This method allows for building a lightweight neural network for real-time applications. However, it ignores the size and the location of the objects and suffers from lots of geometric information losses during quantization. As a result, it is unlikely to exploit sufficient discriminative features for 3D object detection.

2) VOLUMETRIC-BASED METHODS

Volumetric-based methods convert the raw point clouds to standard 3D grids and represents the point clouds as voxels. For instance, *VoxelNet* [21] learns discriminative voxel-wise features for 3D region proposal generation and then proceeds to solve the 3D bounding box regression problem. *SECOND* [22] and *PointPillars* [32] improved *VoxelNet* [21] by proposing an efficient method, named *sparse convolution* [33], to ignore the empty voxels.

3) POINTNETS-BASED METHODS

PointNets [14], [15] models are efficient in the exploitation of point cloud features. *PointRCNN* [20] sets an example in the classification and regression of 3D bounding box directly from dense point clouds. However, the dense processing leads to quite heavy computational costs. A recent one-stage method, *3DSSD* [34] abandons the refinement stage and builds a one-stage anchor-free neural network to directly regress 3D bounding box from the estimated candidate 3D RoIs.

4) POINT-VOXEL METHODS

In order to achieve high detection performance but also to reduce the computational costs, several works [35]–[38] introduced two-stages neural networks for 3D object detection. In the first stage, they coarsely localise the objects and estimate the parameters of the bounding box from the voxel grids generated from the raw point clouds. In the second stage, they introduce a refinement module that leverages the *PointNets* to refine the 3D bounding box.

B. IMAGE-ONLY METHODS FOR 3D OBJECT DETECTION

Wang *et al.* [39] apply LIDAR-only 3D detectors on the Pseudo-LiDAR representations converted from the estimated image-based depth maps. *Stereo R-CNN* [40] applies *Faster R-CNN* [1] on both the left and right images and predicts 3D bounding boxes by learning the projection relations between the associated 2D left-right bounding boxes and 3D bounding box corners.

C. MULTI-SENSOR FUSION METHODS FOR 3D OBJECT DETECTION

In order to leverage the strengths of each sensor, there are several works attempting to fuse point clouds and 2D images

with various strategies. Early works such as *MV3D* [26] and *AVOD* [27] firstly used off-the-shelf 2D feature extractors to capture the feature maps from the images and the multi-view representations of the point clouds (e.g., Bird Eye View and Front View), which are then typically fused together by a sum or a concatenation operation. A Region Proposal Network (RPN) is then applied to the fused feature maps to generate 3D bounding box proposals, followed by a refinement network for final 3D bounding box prediction. The advantages of this method are that mature 2D object detector and 2D feature extractor technologies are available to be applied to the multi-view representations of the point clouds. Furthermore, the features from different sensors can interact over the stacked layers, as these features are normally obtained from similar or even the same neural networks. *Liang et al.* [41] utilizes the continuous convolution method to fuse the feature maps of the images and BEVs. Specifically, this approach proposes a continuous fusion layer that aggregates each pixel feature in the image feature maps with the features of the neighbouring points in the BEV feature maps to learn a fused local region, which allows us to extract sufficient discriminative features for 3D object detection.

In order to narrow the searching space, *Frustum pointnets* [28] and *Frustum convnet* [42] introduced the concept of 3D bounding frustums. The 2D bounding boxes are obtained from mature 2D detectors, and then the 3D frustums are used to trim the point cloud data. Finally, *Pointnets* methods are applied to the trimmed point clouds for carrying out the 3D bounding box regression task. Similarly, *Pointfusion* [29] aggregates the global features of the image obtained from an off-the-shelf 2D feature extractor with the dense semantic features of the point cloud, which are then captured by *Pointnet* [14].

Finally, *PointPainting* [30] densely aggregates the output of the image segmentation neural network with the point clouds before applying LIDAR-only 3D detectors to boost the performance of the 3D object detection task.

III. RoIFusion ARCHITECTURE

In this section, we introduce and describe our RoIFusion neural network for 3D object detection as shown in Fig. 1, which uses both raw point clouds and 2D images as input. Our goal is to leverage the fused information captured from both sensors to classify and localize the objects within the oriented 3D bounding boxes. Three salient points of our model architecture need to be highlighted:

- A fused keypoints generation layer (FKG layer) is proposed, aimed at estimating a set of keypoints from the point clouds and the images.
- A RoIs fusion layer is used to generate 3D/2D RoIs using the obtained keypoints, followed by the 3D/2D RoI pooling operations to obtain the corresponding features to proceed to further RoIs fusion operations.
- A prediction layer is proposed to infer the parameters of the oriented 3D bounding boxes and corresponding classes.

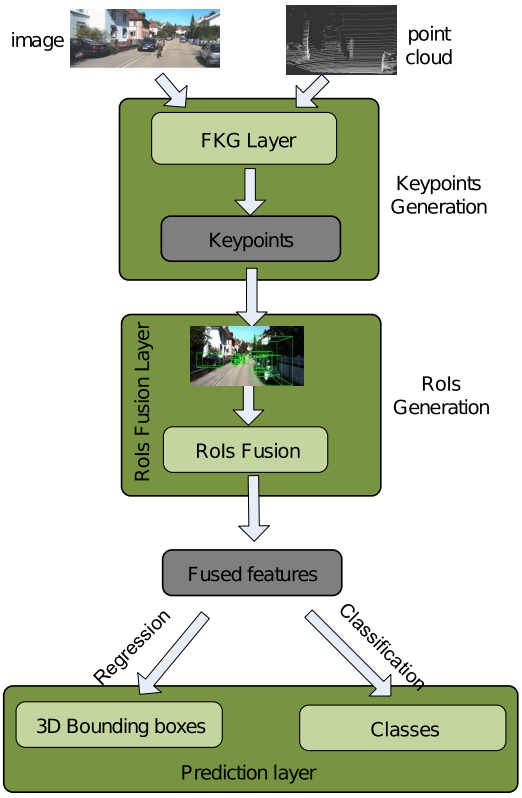


FIGURE 1. Flow chart of the proposed RoIFusion architecture. The flow chart consists of three main steps: keypoints generation, RoIs generation, and 3D bounding boxes prediction. The *FKG* indicates keypoints generation layer.

A. FUSED KEYPOINTS GENERATION (FKG) LAYER

Instead of generating 3D region proposals relying on all the foreground points, we only estimate a small set of 3D keypoints on the objects to generate the RoIs for deep fusion. As illustrated in part (a) of Fig. 2, our FKG layer takes the raw point clouds and the RGB images as input, and combines point-guided keypoints and pixel-guided keypoints. The fused keypoints are generated by leveraging the segmented point-clouds and images. As a result, we obtain a set of keypoints on the objects leveraging both the point cloud and the image information.

We can define the point cloud as shown in Eq. (1), where $\mathbf{x_i}$ denotes the i-th point with the 3D space coordinates $[x_i, y_i, z_i]$ and the measured reflectance r_i. As a result, the dimension of the point cloud data set is $N \times 4$.

$$\mathbf{X} = \left\{ \mathbf{x}_i = [x_i, y_i, z_i, r_i] \in \mathbb{R}^4, i = 1, 2, \ldots, N \right\} \quad (1)$$

1) POINT-GUIDED KEYPOINTS GENERATION

Our point-guided keypoints $\mathbf{X}^{\hat{(pc)}} \in \mathbb{R}^{M_1, F}$, where M_1, F are the numbers of keypoints and corresponding features respectively, are extracted by the set abstraction (SA) layers backbone, as illustrated in part (d) of Fig. 2. This is done as proposed by *PointNet++* [15], where a simultaneous

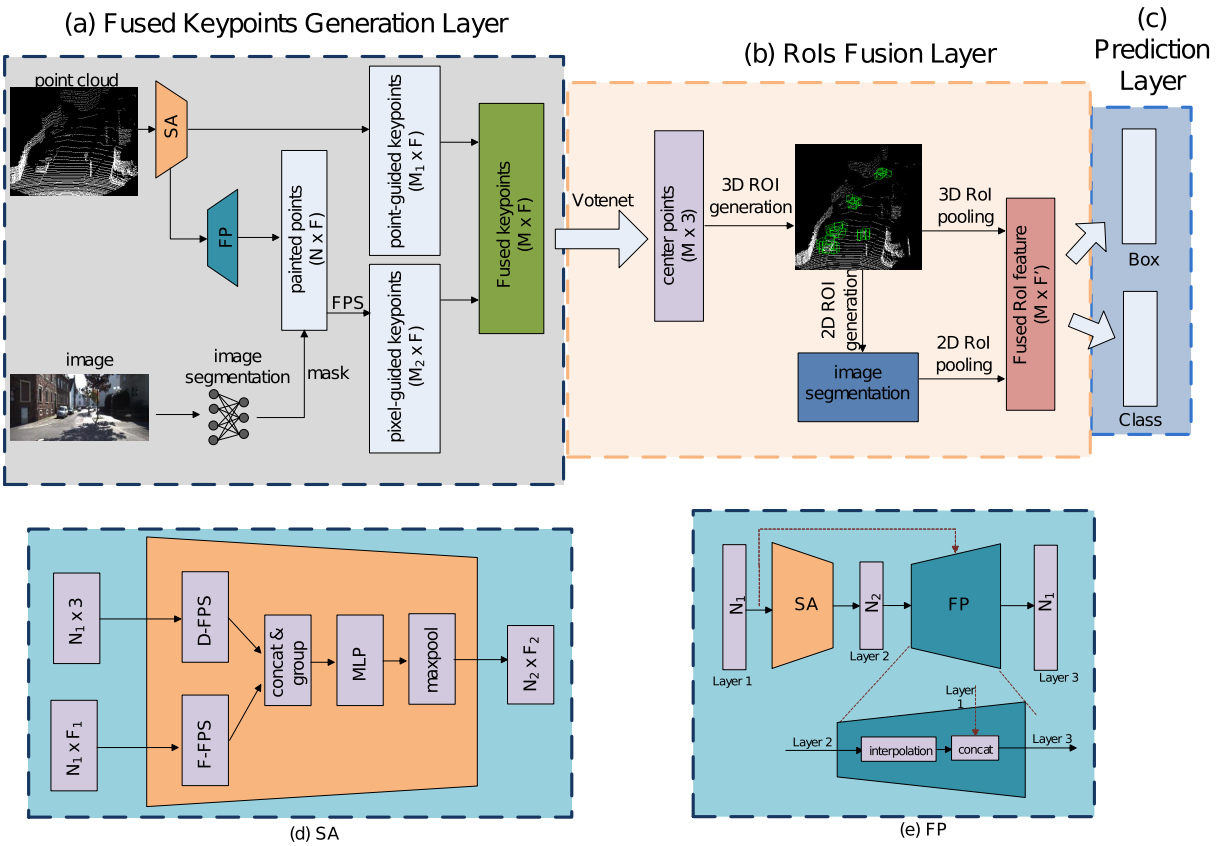


FIGURE 2. RoIFusion framework. As a whole, the proposed architecture contains three parts: (a) a Fused Keypoints Generation (FKG) layer which uses the raw point clouds and the images as input and aggregates the keypoints generated from both the point cloud set abstract (SA) network and the image segmentation network respectively. (b) The keypoints are then used to estimate the center points of the potential objects and the 3D Region of Interests (RoIs), which are projected to the image to obtain the 2D RoIs. (c) The 3D/2D RoI pooling layers are employed to capture the respective local features, which are finally fused together for 3D bounding box prediction. (d) The inputs of the SA module are the 3D points $N_1 \times 3$ and the corresponding features $N_1 \times F_1$. We simultaneously downsample the points, extract the corresponding deep features in the orange block, and obtain the downsampled 3D points $N_1 \times 3$ and corresponding features $N_2 \times F_2$. N_1, F_1, N_2, F_2 are the dimension number of the input points/features, output points/features respectively. (e) Feature Propagation (FP) module: after applying a SA module, the number of the points is downsampled from N_1 to N_2. Successively, the FP module takes layer1 with N_1 points and layer2 with N_2 points as input. After that, the number of points N_2 in the layer2 is firstly interpolated to N_1, the output of which is then concatenated to layer1 to obtain the layer3 with the number of points N_1.

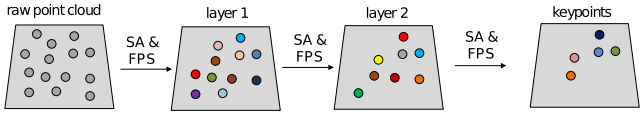


FIGURE 3. Illustration for point-guided keypoints generation.

downsampling of the points and extraction of the corresponding deep features are carried out.

Specifically, as shown in Fig. 3, a certain number of SA layers can be applied, and jointly with a downsampling operation that makes the dimensionality of the raw point cloud reduced. At each layer, a set of points are processed, and a new, smaller set with higher-level features is generated. Finally, in the last layer we obtain an extremely reduced number of points that are used as keypoints.

For what concerns the downsampling strategy, we use an iterative farthest point sampling (FPS) method to select the points of the subset. Let us suppose an empty subset $\mathbf{X1}$, a random point is firstly picked and added to $\mathbf{X1}$, then the point having the farthest 3D geometric Euclidean distance is iteratively added to $\mathbf{X1}$ until the expected M points are picked. This FPS strategy, named *D-FPS*, is characterised by resulting in better coverage of the whole point set than a random sampling strategy. In order to preserve sufficient foreground points and filter out the background, inspired by *3DSSD* [34], we also decided to employ a specific FPS strategy, named *F-FPS*, which calculates the Euclidean distances of the semantic features for the points selection. The *F-FPS* method is beneficial to preserving foreground points (e.g., points on the objects) and removing the useless background, such as points on the ground. Finally, we follow [34] and combine both FPS strategies together to efficiently capture sufficient foreground points as the keypoints.

2) PIXEL-GUIDED KEYPOINTS GENERATION

Considering the fact that colour and texture representations are useful to localize objects within point clouds, especially

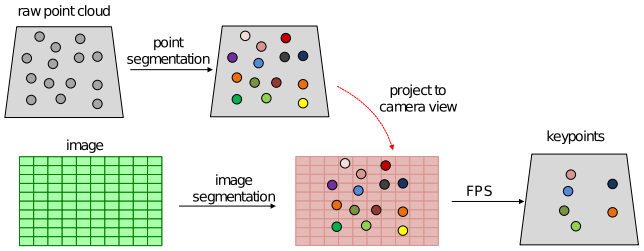


FIGURE 4. Illustration for pixel-guided keypoints generation.

for small objects that are difficult to be detected by LIDAR-only detectors, we capture the segmentation features and corresponding scores using an image segmentation network, which is then used to guide the keypoints selection as shown in Fig. 4.

Algorithm 1 Pixel-Guided Keypoints Generation

Input: Point clouds: $\mathbf{X} \in \mathbb{R}^{N,4}$.
Images: $\mathbf{I} \in \mathbb{R}^{W,H,3}$.
Homogeneous transformation matrix: $\mathbf{T} \in \mathbb{R}^{4,4}$.
Camera projection matrix: $\mathbf{M} \in \mathbb{R}^{3,4}$.
Output: Pixel-guided keypoints features: $\hat{\mathbf{X}}^{(\mathbf{img})} \in \mathbb{R}^{M_2,F}$.

1: Apply point cloud segmentation network to obtain segmentation features: $\mathbf{X_s} \in \mathbb{R}^{F_p}$.
2: Apply image segmentation network to obtain segmentation features: $\mathbf{I_s} \in \mathbb{R}^{W,H,F_i}$ and segmentation scores: $\mathbf{S} \in \mathbb{R}^{W,H,C}$.
3: $\mathbf{X_{img}}$, **indices** $= Projection(\mathbf{M}, \mathbf{T}, \mathbf{X})$.
4: $\mathbf{X_{obj}}$, **indices** $= Mask(\mathbf{X_{img}}, \mathbf{S}, \mathbf{indices})$.
5: $\mathbf{X_s^{(obj)}} = Mapping(\mathbf{X_s}, \mathbf{indices})$.
6: $\hat{\mathbf{X}}^{(\mathbf{img})} = FPS(\mathbf{X_s^{(obj)}})$.
7: **return** $\hat{\mathbf{X}}^{(\mathbf{img})}$.

The detailed algorithm used to extract the pixel-guided keypoints is shown in Alg. 1. Firstly, we generate the segmentation features of the point cloud $\mathbf{X_s}$ using a Feature Propagation (FP) layer as shown in part (e) of Fig. 2. In particular, we leverage the output of the SA layer as input, and upsample the points by interpolating the point features using the inverse squared Euclidean distance weighted average function as shown in Eq. (2). Furthermore, we concatenate the interpolated point features with the skip-linked point features from the corresponding SA layer. As a result, our FP layer outputs the 3D geometric points and corresponding semantic features with the same number of points as the raw point cloud.

$$f(\mathbf{x}) = \frac{\sum_{i=1}^{k} \omega_i(\mathbf{x}) f_i}{\sum_{i=1}^{k} \omega_i(\mathbf{x})} \tag{2}$$

where $\omega_i(\mathbf{x}) = \frac{1}{(\mathbf{x}-\mathbf{x_i})^2}$ is the inverse squared Euclidean distance between a certain point $\mathbf{x}$ and corresponding i-th neighbouring point $\mathbf{x_i}$ of the $k = 3$ nearest neighbours.

For what concerns image processing, it is a common choice in literature to use the mature 2D feature extractors to capture the feature maps from the RGB images. However, these feature maps are unlikely to localize the objects in the images. Consequently, we use a lightweight image segmentation neural network *DeepLabv3* [43] to efficiently capture pixel-wise segmentation features $\mathbf{I_s}$ and segmentation scores $\mathbf{S}$, which allows us to ignore the background and conduct the keypoints selection. It is worth pointing out that our RoIFusion model is agnostic to the development of the image segmentation models.

After that, the *Projection* method (see Step 3 in Alg. 1) projects the point cloud to the image viewpoint using a homogeneous transformation as shown in Equation 3. As a result, the point clouds are painted by the corresponding images with the segmentation scores $\mathbf{S}$ of the relevant pixels. Besides, we also obtain the mapping indices that associate all the points in the point cloud viewpoint with corresponding points in the image viewpoint.

Then the *Mask* method (see Step 4 in Alg. 1) takes the painted point cloud, corresponding scores $\mathbf{S}$ and indices as input. We set a global threshold for all the obtained scores $\mathbf{S}$ to convert the scores to the binary values, which allows us to only select the pixel-guided foreground points and corresponding indices that are associated with the binary value 1.

The *Mapping* method (see Step 5 in Alg. 1) further uses the reserved indices to project these selected points from the image viewpoint back to the point cloud viewpoint. Consequently, these pixel-guided points are also the foreground in the corresponding image, which is beneficial for selecting the foreground points with few geometric features but sufficient colour information.

At last, we use the F-FPS as our downsampling strategy to further select a small set of point segmentation features $\mathbf{X}^{(\hat{img})}$ with the dimension size of $M_2 \times F$, where M_2 and F are the numbers of keypoints and corresponding features respectively.

$$\mathbf{y} = \mathbf{P_{rect}}\ \mathbf{R_{rect}}\ \mathbf{T_{velo}^{cam}}\ \mathbf{x} \tag{3}$$

where $\mathbf{x}$ indicates a certain point in the point cloud, and $\mathbf{P_{rect}}$, $\mathbf{R_{rect}}$ are the projection matrix and rotation matrix respectively. The $\mathbf{T_{velo}^{cam}}$ is the homogeneous matrix for transformation.

3) KEYPOINTS FUSION

The fused keypoints $\hat{\mathbf{X}} \in \mathbb{R}^{M,F}$ are hence obtained by aggregating the point-guided keypoints with the pixel-guided keypoints along with the channel axis of the number of the keypoints, where M is the number of fused keypoints. It is worth highlighting that the points on objects that are difficult to be detected are more likely to be selected due to the fact that a part of the points are captured based on the image segmentation scores.

B. RoIs FUSION LAYER

After the implementation of the FKG layer intertwined with the point cloud segmentation network and the image segmentation network, we obtain a set of keypoints scattered

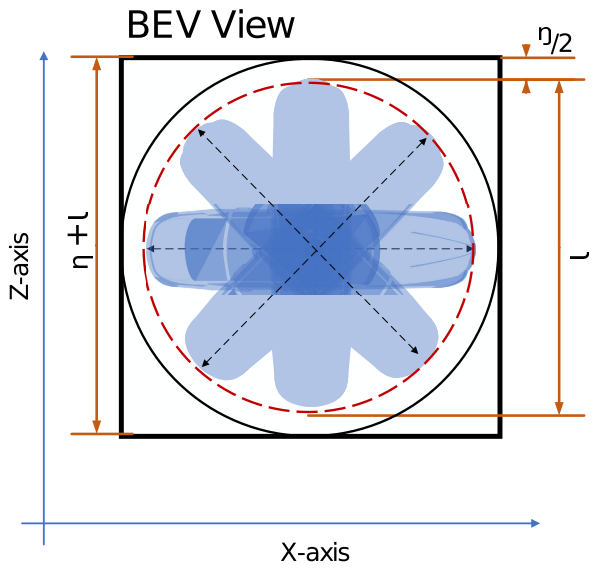


FIGURE 5. RoIs generation layer. The objects with all the oriented angles are illustrated in the red dot circle. The length size of the objects (e.g., cars) are defined as l. η is the enlarged size for the object length. As a result, $\eta + l$ represents the size of the RoI in the Bird eye view (BEV).

over the objects. The keypoints are also used to predict the center of the objects before we generate the RoIs. In fact, considering that these keypoints lie on the objects, inspired by [44], the spatial location and features of the keypoints can be used to estimate the centers of the corresponding objects. As shown in part (b) of Fig. 2, these high-level keypoints are used to generate 3D RoIs in the point cloud view and corresponding 2D RoIs in the camera view, followed by a RoI fusion operation to obtain the fused RoI features. Specifically, we build a subnetwork *VoteNet* with a single layer to learn the spatial offset between predicted center points and the corresponding ground truth. We treat each center point as the centroid of the 3D bounding box of the object.

1) 3D RoIs GENERATION AND POOLING

The 3D RoIs are then generated for the previously estimated center points. Successively, we apply a 3D RoI pooling layer to pool the surrounding points of each center point and learn the local features for those clustered points around each center point.

We encode our RoIs using the axis-aligned 3D bounding boxes. Specifically, the centroid of each RoI is parametrized as $(x^{(c)}, y^{(c)}, z^{(c)})$, whilst the length l and the width w of the RoI are set to the enlarged length size of the objects, in order to cover all the possible orientations of the object as shown in Fig. 5. We finally use an enlarged height of the objects as the height h for each RoI. As a result, the dimension of the RoI is defined as $(x_i^{(c)}, y_i^{(c)}, z_i^{(c)}, \eta + h_i, \eta + w_i, \eta + l_i)$, where η is the parameter for extended size of the RoI.

After that, we shift the points inside each 3D RoI to the relative locations based on the center points to improve the local features learning and then apply a subnetwork equipped with stacked Multi-Layer-Perceptron (MLP) layers on the cluster the points inside the 3D RoIs to extract the local RoI pooling features.

2) 2D RoIs GENERATION AND POOLING

Our 3D RoIs are then projected to the image to generate the corresponding 2D RoIs, followed by a 2D RoI pooling layer, inspired by [1], to learn the local texture features for the 2D RoIs.

3) RoI POOLING FEATURES FUSION

We finally fuse the point cloud 3D RoI and the image 2D RoI by aggregating the pooling features along with feature dimension axis as shown in Fig. 7. Specifically, we define a fusion strategy by concatenation as in Eq. (4):

$$\mathbf{F}_{\mathbf{fuse}} = MLP(concat[\mathbf{F}_{\mathbf{roi}}^{(\mathbf{pc})}, \mathbf{F}_{\mathbf{roi}}^{(\mathbf{img})}]) \tag{4}$$

where $\mathbf{F}_{\mathbf{fuse}}$ is the fused feature from the 3D RoI pooling features $\mathbf{F}_{\mathbf{roi}}^{(\mathbf{pc})}$ and 2D RoI pooling features $\mathbf{F}_{\mathbf{roi}}^{(\mathbf{img})}$.

C. PREDICTION LAYER

The prediction layer, inspired by [34], uses an anchor-free method to directly predict the offset between the center points and the corresponding ground truth of the center of the 3D bounding box for regression. Besides, we also directly regress the 3D bounding box size from the fused RoI features. For the orientation regression, we follow the method introduced in [28] relying on a hybrid classification and regression algorithm to estimate the orientation angle of the 3D bounding box. In particular, we split the orientation into H equal angle bins and use the output of the RoI fusion layer to classify the said angle bins, and then regress residuals with respect to the classified bin.

IV. MODEL STRUCTURE

The model structure is presented in Fig. 2. In our experiments, we firstly sampled randomly $N = 16384$ points from the raw point cloud. Successively, we apply the SA layer Fig. 6(a), the FP layer Fig. 6(b), and the image segmentation network *DeepLabv3* to capture $M = 256$ keypoints. The hyper-parameters of the SA layer and the FP layer are represented in Fig. 6(a) and Fig. 6(b) respectively. Successively, we employ a single layer *Votenet* with filters (128) to estimate the center points for the 3D bounding box of the objects. The dimensions of the 3D RoIs are set to $[h = 1.8\ m, w = 5.0\ m, l = 5.0\ m]$, $[h = 1.8\ m, w = 1.0\ m, l = 1.0\ m]$, $[h = 1.8\ m, w = 1.8\ m, l = 1.8\ m]$ for the car, pedestrians, and cyclists objects respectively. We set the constant extended value $\eta = 1.0\ m$. We finally set the number of the angle bins to $H = 12$.

V. EXPERIMENTS

In this section, we evaluate our deep fusion method on the widely used KITTI 3D object detection benchmark [48], [49]. We firstly introduce the KITTI dataset and explain the

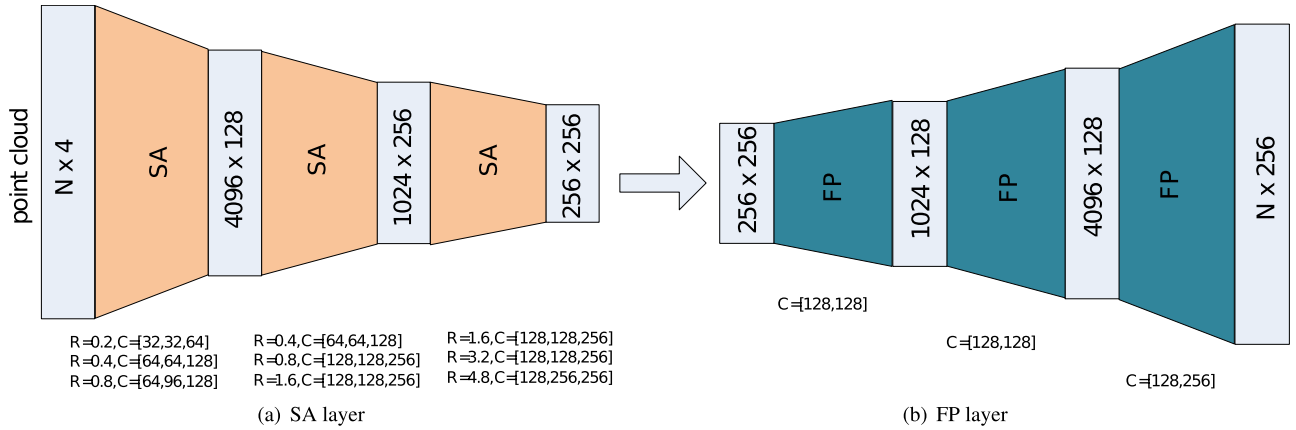


FIGURE 6. Illustration of the SA and FP layers. The SA layer 6(a) takes *N* points as input, followed by stacked SA layers to downsample points and extract corresponding features. After that, the FP layer 6(b) upsamples the points to the original 16384 points using stacked FP layers. *R* is the radius of the ball query strategy for clustering local region points, and *C* is the number of the filters for the MLP layers.

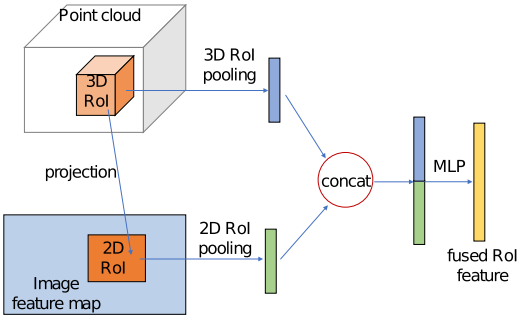


FIGURE 7. RoIs fusion layer.

detailed training settings. Then, we compare our results with recent state-of-the-art 3D detectors. We only test our model on the car category due to a large amount of data after preprocessing. However, we evaluate all the categories when we compare our model to the backbone model *3DSSD* [46] to present the effectiveness of our fusion method. Finally, we analyse the efficiency of our fusion method and visualize some representative results for our 3D object detection model.

A. DATASET

The KITTI dataset [48] contains both 2D images and 3D point clouds with the corresponding annotations for the cars, pedestrians, and cyclists categories in an urban driving scenario. The sensors used for data collection are: 2 grayscale cameras, 2 colour cameras, and 1 Velodyne HDL-64E LIDAR. We only used the point clouds data and images from the left colour camera to train our fusion model. The dataset provides 7481 samples for training and 7518 samples for testing. As a standard good practice, we further split the KITTI training dataset into 3712 samples for training and 3769 samples for validation. We evaluated our model on the validation dataset following the easy, moderate, and hard difficulty classification levels officially introduced by KITTI. Specifically, in order to align the performance of the algorithms and cover most of the traffic scene scenarios, the object detection task is divided into three levels for validation and testing with respect to the different size, occlusion, and truncation level as shown in Table 2. Besides, the average precision (AP) metric is used when we compare our results with other different models.

B. IMPLEMENTATION DETAILS

We used the Adam [50] algorithm as our training optimizer. The batch size was set to 4 on an NVIDIA 1080Ti GPU. The learning rate was initially set to 0.002, and then was divided by 10 at 40 epochs. Our model has been trained for a total of 50 epochs.

During training processing, we introduce two data augmentation methods on the original KITTI dataset. Specifically, we firstly separate the foreground objects with the corresponding inner points out and then randomly add them into the current point cloud. Then we augment the training dataset by rotating the orientation of the bounding boxes in a uniform distribution and translating the center of the bounding boxes in random.

Wit respect to the testing processing, the batch size is set to 1, and threshold of the classification confidence is set to 0.3.

C. RESULTS

As shown in Table 1, for the 3D car detection, our model also achieves the best performance compared to recent state-of-the-art fusion methods on the test dataset. We choose *moderate* difficulty as the main average precision (AP) metric, and compare our model to BEV-image fusion *MV3D* [26] and *AVOD* [27], our deep fusion method outperforms all others by a large margin. For the frustum method, our method outperforms *F-PointNet* [28] by 7.12%. Besides, our model significantly outperforms point-pixel-wise fusion method *Painted PointRCNN* [30] by 6.21%. We also visualize some examples for prediction results and corresponding ground truth as shown in Fig. 9 for better representation. However, the performance of our fusion method for 3D car detection

TABLE 1. 3D car detection results on KITTI test dataset. *Sen.* indicates involved sensors by the methods. *L* and *I* denote LIDAR and images respectively.

Method	Sen.	$AP_{Car}(\%)$			$AP_{Ped.}(\%)$			$AP_{Cyc.}(\%)$		
		easy	mod.	hard	easy	mod.	hard	easy	mod.	hard
VoxelNet [21]	L	77.47	65.11	57.73	39.48	33.69	31.51	61.22	48.36	44.37
SECOND [22]		83.13	73.66	66.20	51.07	42.56	37.29	70.51	53.85	46.90
PointPillars [32]		79.05	74.99	68.30	52.08	**43.53**	**41.49**	75.78	59.07	52.92
PointRCNN [20]		85.94	75.76	68.32	47.98	39.37	36.01	74.96	58.82	52.53
Fast PointRCNN [36]		84.28	75.73	67.39	-	-	-	-	-	-
Patches [45]		87.87	77.16	68.91	-	-	-	-	-	-
Part A2 [35]		85.94	77.86	72.00	53.10	43.35	40.06	**79.17**	**63.52**	**56.93**
STD [46]		86.61	77.63	**76.06**	**53.29**	42.47	38.35	78.69	61.59	55.30
3DSSD [34]		**88.36**	**79.57**	74.55	-	-	-	-	-	-
MV3D [26]	L+I	71.09	62.35	55.12	-	-	-	-	-	-
AVOD [27]		81.94	71.88	66.38	46.35	39.00	36.58	59.97	46.12	42.36
F-PointNet [28]		81.20	70.39	62.19	51.21	**44.89**	40.23	71.96	56.77	50.39
F-ConvNet [42]		85.88	76.51	68.08	52.16	43.38	38.80	**81.98**	**65.07**	56.54
IPOD [47]		79.75	72.57	66.33	**56.92**	44.68	**42.39**	71.40	53.46	48.34
Painted PointRCNN [30]		82.11	71.70	67.08	50.32	40.97	37.87	77.63	63.78	55.89
Ours		**88.32**	**79.54**	**74.47**	42.22	35.14	32.92	80.84	64.05	**58.37**

TABLE 2. KITTI dataset difficulty classification levels for object detection.

Levels	Min. bounding box height	Max. occlusion level	Max. truncation
Easy	40 pixels	Fully visible	15%
Moderate	25 pixels	Partly occluded	30%
Hard	25 pixels	Difficult to see	50%

is similar to our backbone model *3DSSD* [46], and we discuss the reason that the results for each experiment have fluctuations due to the random downsampling processing in the model. Consequently, the model is actually applied on the different points during inference, which leads to different results.

However, it is easy to observe that our model performs the worst for pedestrian detection when comparing to other fusion methods [27], [28], [47]. We discuss that two-stage detection methods, which include coarse detection module and refinement module, normally perform better than one-stage. Specifically, a large number of region proposals firstly are generated in the coarse detection module, followed by the refinement module for further finer estimation, which is more likely to capture the small objects (e.g., pedestrians). Besides, our one-stage method directly generates a small set of keypoints from a large amount of points, which leads to the fact that it is still difficult to select the small objects that only contain quite few points. Last but not least, although our fusion model could capture the points on the small objects using the pixel information, the fused features from these points and corresponding pixels are still not sufficient to regress the 3D bounding boxes. With regard to the 3D cyclist detection, our model achieves competitive performance compared to other state-of-the-art fusion methods.

We also compare our model to the backbone network *3DSSD* [46] on the validation dataset to show the effectiveness of our fusion strategy as shown in Table 3. The bottom line indicates the difference between our model and *3DSSD* for 3D car detection. It shows that our model outperforms *3DSSD* in all the categories and all the difficulty levels, which convincingly shows the efficiency of our RoI fusion method. Take the moderate difficulty performance level on the validation dataset as an example, our fusion method significantly improves 3DSSD model by 3.29%, 1.93%, 1.73% for car, pedestrian and cyclist detection respectively. Besides, as shown in Table 3, we further introduce the average results in all the difficulty classification levels performance for car, pedestrian and cyclist objects detection, and it demonstrates that our model outperforms *3DSSD* by 2.14%. As a result, the results verify the effectiveness of our fusion method.

In order to show the reproducibility of our model, we statistically perform our model in the same computing environment 10 times and then calculate the standard deviation of the moderate results for the 3D car detection as shown in Table 4.

D. ABLATION STUDY

We carried out several ablation experiments to investigate the effectiveness of extended value for RoI size and different fusion methods. All the experiments are performed on the KITTI validation dataset for the 3D car detection task.

1) EFFECTS OF THE EXTENDED SIZE OF RoI

In order to increase the number of selected points around the objects, we can enlarge the RoI size by an extended

TABLE 3. 3D Car detection average precision (AP) on KITTI validation dataset compared to *3DSSD* model. The *Delta* indicates the difference between our model and *3DSSD* model that is our backbone for the point cloud processing. *Avg* is the average accuracy in all the difficulty classification levels for the car, pedestrian and cyclist detection. *repro* represents that the results are reproduced on our own computer.

Method	Avg	$AP_{car}\%$			$AP_{ped}\%$			$AP_{cyc}\%$		
		easy	mod.	hard	easy	mod.	hard	easy	mod.	hard
AVOD [27]	57.49	81.94	71.88	66.38	50.80	42.81	40.88	64.00	52.18	46.61
F-PointNet [28]	58.8	81.2	70.39	62.19	51.21	44.89	40.23	71.96	56.77	50.39
IPOD [47]	69.54	84.1	76.4	75.3	69.6	62.3	64.6	81.9	57.1	54.6
3DSSD (repro) [34]	68.90	89.71	79.45	78.67	63.05	58.04	53.86	78.01	62.32	57.01
Ours	71.04	91.36	82.74	80.22	66.21	59.97	55.64	80.84	64.05	58.37
Delta	2.14	+1.65	+3.29	+1.55	+3.16	+1.93	+1.78	+2.83	+1.73	+1.36

TABLE 4. The standard deviation of the reproducibility of the result. *Dev* indicates the standard deviation of the results for 10 evaluation experiments.

Experiments	Dev	01	02	03	04	05	06	07	08	09	10
Ours	0.0926	82.53%	82.65%	82.73%	82.74%	82.46%	82.60%	82.67%	82.70%	82.69%	82.56%

TABLE 5. Effectiveness of the extended value η for RoI size on KITTI car validation dataset.

η (m)	$AP_{easy}(\%)$	$AP_{mod.}(\%)$	$AP_{hard}(\%)$
0.0	89.33	80.61	70.47
0.5	91.12	82.53	79.85
1.0	**91.36**	**82.74**	**80.22**
1.5	90.42	82.48	79.11
2.0	89.01	81.09	78.96

TABLE 6. Effectiveness of the fusion strategy on KITTI car validation dataset.

fusion strategy	$AP_{easy}(\%)$	$AP_{mod.}(\%)$	$AP_{hard}(\%)$
sum	88.29	76.72	68.48
concat	**91.36**	**82.74**	**80.22**
max	87.01	74.93	67.27

TABLE 7. Inference time, the parameters and accuracy on the *moderate* level for different fusion methods. *para.* indicates the number of the parameters (Million).

	AVOD [27]	F-PointNet [28]	PointPainting [30]	Ours
time	100 ms	170 ms	400 ms	220 ms
para.	38.07 M	1.9 M	-	3.48 M
AP	71.48%	70.39%	71.70%	79.54%

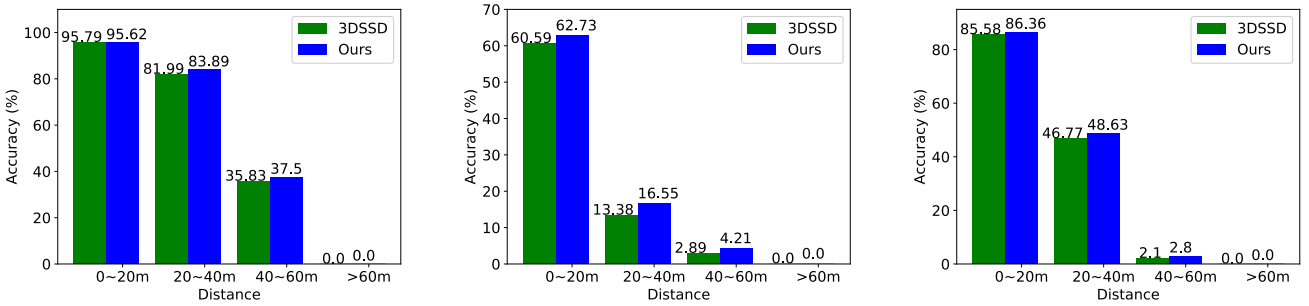
value η, and evaluate the effect using more contextual local features. Table 5 shows that the model achieves the best performance when $\eta = 1.0$. Besides, we notice that there is a significant drop in performance when no extended size is assumed (i.e. $\eta = 0$), especially for the hard difficulty level of detection. In fact, in this case, typical objects are either partially occluded by other objects or far away from the sensor, which leads to having few points per objects. As a result, involving more surrounding points is typically beneficial to object classification and regression. On the other hand, when a larger size of the box is assumed (i.e. $\eta = 2$), sufficient information is provided to the network, but jointly to redundant and harmful information leading to a decrease of performance. In contrast, smaller values of η only could provide partial information of the objects (i.e. cars), which is insufficient to predict the parameters of the 3D bounding box.

2) EFFECTS OF THE FUSION STRATEGY

We further investigate the effectiveness of the different fusion strategies. In addition to the concatenation operation as described in Section III-B3, we also employ operations, such as sum, max operation, and compare the results for different choices. As shown in Table 6, the concatenation operation for RoI features fusion achieves 91.36% 82.74% 80.22% performance for easy, moderate, and hard difficulties, respectively. The results show that the concatenation operation could fuse more discriminative features from the 3D RoIs and corresponding 2D RoIs. This can be linked to the fact that both the sum operation and the max operation could obtain signature features, but the concatenation operation allows to keep all the features from different sensors, which then is likely to allow capturing more useful features for classification and regression.

3) EFFECTS OF THE DISTANCE OF THE OBJECTS

As we previously discussed, detecting small-objects or far away objects relying on point cloud data only is often difficult due to the sparsity of the information. In order to present the efficiency for our fusion method, we only compare the difference to our backbone LIDAR-only model *3DSSD* [46] for the detection performance in the various distance ranges. As illustrated in Fig. 8, it is easy to observe that our results only have slight differences with the *3DSSD* when detecting the cars within 20 meters. We discuss that LIDAR-only methods are likely to obtain sufficient features from the



(a) Car Detection (b) Pedestrian Detection (c) Cyclist Detection

FIGURE 8. The comparison of the accuracy performance for the detected objects in the various distance ranges.

FIGURE 9. Qualitative results on the KITTI validation dataset. The predicted objects and the ground truth objects are shown in red and green bounding boxes respectively. We also project the bounding boxes to the RGB images for better visualization.

points on the nearby cars. However, for objects at distances greater than 20 meters, the performance decreases for both models, but our fusion-based model performs significantly better than *3DSSD* when detecting the objects located in the long distance. Besides, our fusion method remarkably outperforms *3DSSD* for the pedestrians and cyclists detection in all the distance ranges, although the performance dramatically decreases when the pedestrians and cyclists located in the long-distance (e.g., greater than 40 meters). As a result, these results convincingly show that our fusion method has higher accuracy in the long-distance range as a consequence of learning more discriminative colour and texture information than having to rely only on the point cloud information. It is worthwhile to mention that the improvement is still restricted for the small object detection (e.g., pedestrians, cyclist) due to the fact that it is still difficult to regress the parameters of the 3D bounding boxes leveraging quite a few points information, although the pixel features are beneficial for locating the foreground points on the small objects.

4) INFERENCE TIME

We tested the inference time on KITTI validation dataset with a NVIDIA 1080Ti GPU, and then compare to existing fusion methods in Table 7. Our model achieves the best trade-off compared to BEV-image fusion method *AVOD* [27] and frustum method *F-PointNet* [28]. We also note that our sparse

RoI fusion method is much better than dense point-pixel fusion method *PointPainting* [30] in terms of both accuracy and inference time.

E. QUANTITATIVE ANALYSIS

The comparison of our model with other multi-sensor methods, as shown in Table 1, shows that our model achieves higher efficiency and effectiveness. The RoI fusion enables to globally fuse the local area from the point clouds and the images, making it easier to align the viewpoint when we concatenate the 3D/2D RoI pooling features. In contrast, the point-pixel fusion is unlikely to obtain discriminative features due to the fact that the point clouds are sparse and irregular, conversely to images, which are always conforming to a standard grid-like structure. Compared to BEV-image fusion, our model outperforms others by a large margin, probably due to the information losses during the BEV generation.

VI. CONCLUSION

In this paper, we proposed a novel deep fusion method, named RoIFusion, which efficiently fuses 3D-point cloud data and 2D-images for carrying out 3D object detection for autonomous vehicle navigation. We proposed a lightweight neural network able to generate 3D RoIs from the point clouds and 2D RoIs from the images, and then employing a 3D RoI pooling layer and a 2D RoI pooling layer in order to obtain the geometric features and the texture features, respectively. Finally, we fuse the extracted features together to predict the oriented 3D bounding box for the final detected object. Our fusion method is flexible and could combine any other LIDAR-only segmentation network and/or image segmentation network. The state-of-the-art performance of our model convincingly shows that the fusion method proposed can successfully boost the performance of 3D object detection.

REFERENCES

[1] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 91–99.

[2] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.

[3] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 21–37.

[4] J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: Object detection via region-based fully convolutional networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 379–387.

[5] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2117–2125.

[6] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2980–2988.

[7] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, "CenterNet: Keypoint triplets for object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 6569–6578.

[8] B. Xue and N. Tong, "DIOD: Fast and efficient weakly semi-supervised deep complex ISAR object detection," *IEEE Trans. Cybern.*, vol. 49, no. 11, pp. 3991–4003, Nov. 2019.

[9] B. Xue and N. Tong, "Real-world ISAR object recognition using deep multimodal relation learning," *IEEE Trans. Cybern.*, vol. 50, no. 10, pp. 4256–4267, Oct. 2020.

[10] D. Hong, L. Gao, N. Yokoya, J. Yao, J. Chanussot, Q. Du, and B. Zhang, "More diverse means better: Multimodal deep learning meets remote-sensing imagery classification," *IEEE Trans. Geosci. Remote Sens.*, early access, Aug. 24, 2020, doi: 10.1109/TGRS.2020.3016820.

[11] D. Hong, N. Yokoya, G.-S. Xia, J. Chanussot, and X. X. Zhu, "X-ModalNet: A semi-supervised deep cross-modal network for classification of remote sensing data," *ISPRS J. Photogramm. Remote Sens.*, vol. 167, pp. 12–23, Sep. 2020.

[12] D. Hong, N. Yokoya, N. Ge, J. Chanussot, and X. X. Zhu, "Learnable manifold alignment (LeMA): A semi-supervised cross-modality learning framework for land cover and land use classification," *ISPRS J. Photogramm. Remote Sens.*, vol. 147, pp. 193–205, Jan. 2019.

[13] A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet classification with deep convolutional neural," in *Proc. Neural Inf. Process. Syst.*, 2014, pp. 1–9.

[14] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 652–660.

[15] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5099–5108.

[16] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph CNN for learning on point clouds," *ACM Trans. Graph.*, vol. 38, no. 5, pp. 1–12, Nov. 2019.

[17] Y. Liu, B. Fan, S. Xiang, and C. Pan, "Relation-shape convolutional neural network for point cloud analysis," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 8895–8904.

[18] C. Chen, L. Zanotti Fragonara, and A. Tsourdos, "GAPNet: Graph attention based point neural network for exploiting local feature of point cloud," 2019, *arXiv:1905.08705*. [Online]. Available: http://arxiv.org/abs/1905.08705

[19] C. Chen, L. Zanotti Fragonara, and A. Tsourdos, "Go wider: An efficient neural network for point cloud analysis via group convolutions," *Appl. Sci.*, vol. 10, no. 7, p. 2391, Apr. 2020.

[20] S. Shi, X. Wang, and H. Li, "PointRCNN: 3D object proposal generation and detection from point cloud," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 770–779.

[21] Y. Zhou and O. Tuzel, "VoxelNet: End-to-end learning for point cloud based 3D object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4490–4499.

[22] Y. Yan, Y. Mao, and B. Li, "SECOND: Sparsely embedded convolutional detection," *Sensors*, vol. 18, no. 10, p. 3337, Oct. 2018.

[23] B. Yang, W. Luo, and R. Urtasun, "PIXOR: Real-time 3D object detection from point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7652–7660.

[24] M. Simon, S. Milz, K. Amende, and H.-M. Gross, "Complex-YOLO: An Euler-region-proposal for real-time 3D object detection on point clouds," in *Proc. Eur. Conf. Comput. Vis.* Springer, 2018, pp. 197–209.

[25] J. Beltran, C. Guindel, F. M. Moreno, D. Cruzado, F. Garcia, and A. De La Escalera, "BirdNet: A 3D object detection framework from LiDAR information," in *Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2018, pp. 3517–3523.

[26] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3D object detection network for autonomous driving," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1907–1915.

[27] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander, "Joint 3D proposal generation and object detection from view aggregation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 1–8.

[28] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas, "Frustum PointNets for 3D object detection from RGB-D data," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 918–927.

[29] D. Xu, D. Anguelov, and A. Jain, "PointFusion: Deep sensor fusion for 3D bounding box estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 244–253.

[30] S. Vora, A. H. Lang, B. Helou, and O. Beijbom, "PointPainting: Sequential fusion for 3D object detection," 2019, *arXiv:1911.10150*. [Online]. Available: http://arxiv.org/abs/1911.10150

[31] J. Zhou, X. Tan, Z. Shao, and L. Ma, "FVNet: 3D front-view proposal generation for real-time object detection from point clouds," 2019, *arXiv:1903.10750*. [Online]. Available: http://arxiv.org/abs/1903.10750

[32] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "PointPillars: Fast encoders for object detection from point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 12697–12705.
[33] B. Graham and L. van der Maaten, "Submanifold sparse convolutional networks," 2017, *arXiv:1706.01307*. [Online]. Available: http://arxiv.org/abs/1706.01307
[34] Z. Yang, Y. Sun, S. Liu, and J. Jia, "3DSSD: Point-based 3D single stage object detector," 2020, *arXiv:2002.10187*. [Online]. Available: http://arxiv.org/abs/2002.10187
[35] S. Shi, Z. Wang, J. Shi, X. Wang, and H. Li, "From points to parts: 3D object detection from point cloud with part-aware and part-aggregation network," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Feb. 28, 2020, doi: 10.1109/TPAMI.2020.2977026.
[36] Y. Chen, S. Liu, X. Shen, and J. Jia, "Fast point R-CNN," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9775–9784.
[37] Z. Liu, X. Zhao, T. Huang, R. Hu, Y. Zhou, and X. Bai, "TANet: Robust 3D object detection from point clouds with triple attention," 2019, *arXiv:1912.05163*. [Online]. Available: http://arxiv.org/abs/1912.05163
[38] S. Shi, C. Guo, L. Jiang, Z. Wang, J. Shi, X. Wang, and H. Li, "PV-RCNN: Point-voxel feature set abstraction for 3D object detection," 2019, *arXiv:1912.13192*. [Online]. Available: http://arxiv.org/abs/1912.13192
[39] Y. Wang, W.-L. Chao, D. Garg, B. Hariharan, M. Campbell, and K. Q. Weinberger, "Pseudo-LiDAR from visual depth estimation: Bridging the gap in 3D object detection for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 8445–8453.
[40] P. Li, X. Chen, and S. Shen, "Stereo R-CNN based 3D object detection for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 7644–7652.
[41] M. Liang, B. Yang, S. Wang, and R. Urtasun, "Deep continuous fusion for multi-sensor 3D object detection," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 641–656.
[42] Z. Wang and K. Jia, "Frustum ConvNet: Sliding frustums to aggregate local point-wise features for amodal 3D object detection," 2019, *arXiv:1903.01864*. [Online]. Available: http://arxiv.org/abs/1903.01864
[43] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," 2017, *arXiv:1706.05587*. [Online]. Available: http://arxiv.org/abs/1706.05587
[44] Z. Ding, X. Han, and M. Niethammer, "VoteNet: A deep learning label fusion method for multi-atlas segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent.* Cham, Switzerland: Springer, 2019, pp. 202–210.
[45] J. Lehner, A. Mitterecker, T. Adler, M. Hofmarcher, B. Nessler, and S. Hochreiter, "Patch refinement–localized 3D object detection," 2019, *arXiv:1910.04093*. [Online]. Available: http://arxiv.org/abs/1910.04093
[46] Z. Yang, Y. Sun, S. Liu, X. Shen, and J. Jia, "STD: Sparse-to-dense 3D object detector for point cloud," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1951–1960.
[47] Z. Yang, Y. Sun, S. Liu, X. Shen, and J. Jia, "IPOD: Intensive point-based object detector for point cloud," 2018, *arXiv:1812.05276*. [Online]. Available: http://arxiv.org/abs/1812.05276
[48] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231–1237, Sep. 2013.
[49] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 3354–3361.
[50] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980

CAN CHEN is currently pursuing the Ph.D. degree with the School of Aerospace, Transport and Manufacturing, Cranfield University. His research interests include deep learning, point cloud analysis, object detection, and autonomous driving.

LUCA ZANOTTI FRAGONARA has been a Researcher and a Senior Lecturer with the Centre of Autonomous and Cyber-Physical Systems and the Aerospace Integration Research Centre, Cranfield University, since 2016. Since January 2019, he has been the Course Director for the M.Sc. in applied artificial intelligence. He has been a Co-I or contact person in a series of high-impact European research projects in the area of SHM and robotic inspections (FP7-Series, FP7-Unplugged, FP7-Fabric, H2020 FET-OPEN CompInnova, H2020 ITN XP-Resilience) and competitive funded research projects in autonomous inspections using robots and drones of aerospace structures to structural health monitoring for increasing the resilience of industrial plants. He is currently a Supervisor for nine Ph.D. students. He has coauthored two book chapters, more than 40 conference proceedings, and more than 30 articles in international peer-reviewed journals. His research interests include structural health monitoring, artificial intelligence, perception for autonomous systems, and system identification.

ANTONIOS TSOURDOS was appointed as the Head of the Autonomous and Cyber-Physical Systems Centre, in 2007, and the Director of Research for the School of Aerospace, Transport and Manufacturing, in 2015. He is currently a Professor of autonomous systems and control with Cranfield University. He supervised over 30 Ph.D. students. He has coauthored five books, 15 book chapters, and more than 100 articles in international top rated journals. He has also been engaged in research on guidance, control and navigation for single and multiple vehicles, as well as verifiable autonomy of unmanned systems and lately dealing with the newly-important subjects of integrated system health management and cyber-physical systems. He has been involved in a number of research projects on UTM, including ASTRAEA, U-Space funded project EuroDrone, and DfT funded project UTM Open Access and EPSRC funded project CASCADE. He is also a member of the IET Executive Team on Robotics and Automation and the UK Autonomous Systems National Technical Committee. He was a member of the Team Stellar, the winning team for the UK MoD Grand Challenge to develop autonomous vehicles to survey an area, in 2008, and the IET Innovation Award (Category Team, 2009). He is also the Chair of IFAC Technical Committee on Aerospace Control. He is also an editorial board member on several international publications, including *IMechE Journal of Aerospace Engineering*, the IEEE TRANSACTIONS ON INSTRUMENTATION AND MEASUREMENT, and the IEEE TRANSACTIONS ON AEROSPACE AND ELECTRONIC SYSTEMS.