Received June 18, 2019, accepted July 1, 2019, date of publication July 5, 2019, date of current version July 24, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2927012

MCF3D: Multi-Stage Complementary Fusion for Multi-Sensor 3D Object Detection

JIARONG WANG[1,2,3], MING ZHU[1], DEYAO SUN[1,2], BO WANG[1,2], WEN GAO[1], AND HUA WEI[1,2]
[1]Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, Changchun 130033, China
[2]University of Chinese Academy of Sciences, Beijing 100049, China
[3]Changchun University of Science and Technology, Changchun 130022, China

Corresponding author: Ming Zhu (zhu_mingca@163.com)

ABSTRACT We present MCF3D, a multi-stage complementary fusion three-dimensional (3D) object detection network for autonomous driving, robot navigation, and virtual reality. This is an end-to-end learnable architecture, which takes both LIDAR point clouds and RGB images as inputs and utilizes a 3D region proposal subnet and second stage detector(s) subnet to achieve high-precision oriented 3D bounding box prediction. To fully exploit the strength of multimodal information, we design a series of fine and targeted fusion methods based on the attention mechanism and prior knowledge, including "pre-fusion," "anchor-fusion," and "proposal-fusion." Our proposed RGB-Intensity form encodes the reflection intensity onto the input image to strengthen the representational power. Our designed proposal-element attention module allows the network to be guided to focus more on efficient and critical information with negligible overheads. In addition, we propose a cascade-enhanced detector for small classes, which is more selective against close false positives. The experiments on the challenging KITTI benchmark show that our MCF3D method produces state-of-the-art results while running in near real-time with a low memory footprint.

INDEX TERMS 3D object detection, multi-sensor fusion, attention mechanism, autonomous driving.

I. INTRODUCTION

Three-dimensional (3D) object detection predicts the 3D bounding boxes and class labels of objects of interest in a scene, and plays an important role in intelligent robotics perception systems such as autonomous vehicles and drones. Compared to two-dimensional (2D) object detection, richer input data and superior algorithms are required to recover the six-degree-of-freedom (DoF) poses and 3D bounding box dimensions of objects. Recent autonomous vehicles are commonly equipped with both cameras and LIDAR. This is advantageous for exploiting different sensors to obtain information on multiple modalities for accurate and reliable 3D object detection.

3D point clouds generated by LIDAR contain accurate depth and reflection intensity information, and this has been combined with powerful deep learning methods and widely applied in 3D object detection. Related approaches either convert point clouds into 2D bird's-eye view (BEV) images [1]–[4], front view images [5], or structured voxel grid representations [6], [7], or directly take raw discrete and unordered point clouds as inputs to estimate 3D bounding boxes [8]. However, these approaches are ineffective when faced with small or occluded objects, owing to the sparsity of the point clouds. On the other hand, 2D images generated by cameras can provide rich and dense texture descriptions for 3D scenes, but the depth information required for precise 3D localization is difficult to obtain. Whether using monocular [9] or stereo [10], [11] camera approaches, the accuracy of the estimated depth cannot be guaranteed, particularly for unseen or dark scenes. Therefore, several approaches have attempted to combine the advantage of images and point clouds to achieve an accurate and robust perception. In [12]–[15], high-resolution images are utilized to generate proposals while point clouds are applied to estimate 3D bounding boxes. However, these cascade networks do not have the capability to perform joint reasoning on multimodal inputs, and the final detection results are largely influenced by the performance of the 2D image-only detection network.

Other approaches [16]–[19] utilize the LIDAR BEV with an image to generate 3D proposals to estimate the final oriented 3D bounding boxes. Reference [18], [19] fuse the

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

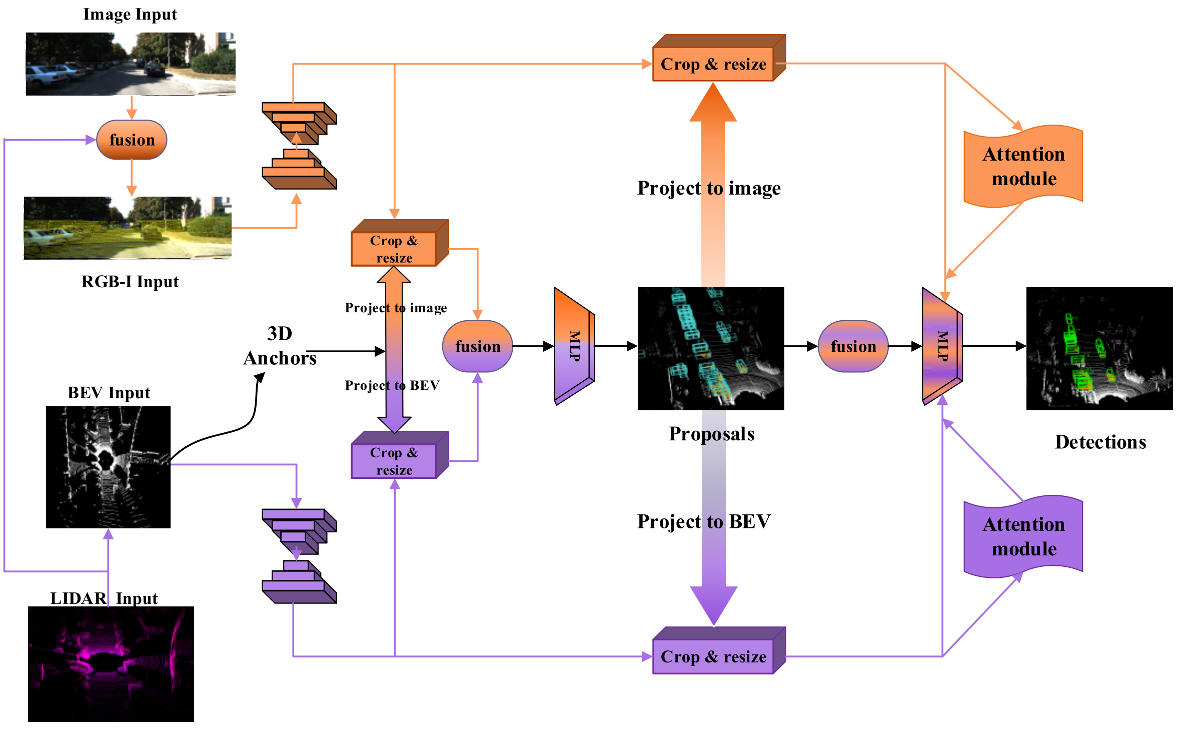

FIGURE 1. **Multi-stage complementary fusion 3D object detection network (MCF3D): The network consists of a 3D RPN and a second-stage detector(s) subnet. The first stage produces non-oriented region proposals by fusing anchor-dependent features from LIDAR and camera views. The second stage jointly predicts object class and do oriented 3D box regression by fusing proposal-dependent features from both views. We design different fusion methods for each stage according to the corresponding tasks, including "pre-fusion," "anchor-fusion," and "proposal-fusion."**

whole feature maps from various views before the region proposal stage. These global-based fusion methods are time-consuming and prone to redundant data. MV3D [16] and AVOD [17] combine region-based multimodal features at the region proposal network (RPN) or detection stage. They usually deal with different view features in a proportional and rigid manner, such as by concatenating or averaging them, using fully connected layers as required. This is clearly inefficient, in the same way that humans do not confuse color perception with geometric spatial perception when dealing with visual problems. LIDAR and image modalities are complementary and parallel, and their weights should be adaptive to various appearances instead of fixed. For example, when faced with small or similarly shaped objects, detection should heavily rely on dense RGB and texture information from camera images, owing to the sparsity of point clouds. Conversely, when the geometric spatial information from point clouds is sufficient to represent the objects, adding extra RGB information will result in a detection performance degradation.

In this paper, we address the above challenges by proposing MCF3D, a multi-sensor 3D object detection method based on multi-stage complementary fusion, which maximizes the strength of multimodal information via fine and targeted fusion. As illustrated in Fig. 1, MCF3D is an end-to-end learnable architecture consisting of a 3D region proposal subnet (3D RPN) and a second stage detector(s) subnet. First, 2D images and LIDAR point clouds are preprocessed as inputs, and CNNs are applied to these to extract high-resolution feature maps. Second, 3D anchors are generated from BEV, and anchor-dependent features from different views are fused to produce 3D non-oriented region proposals. Finally, the proposal-dependent features are fused and passed to the detection subnet for dimension refinement, orientation estimation, and category classification. We design a series of multimodal fusion methods for the tasks of each of the above stages: (1) "Pre-fusion" is performed at the input preprocessing stage, using spatial geometric constraints and prior knowledge to selectively match and transform the camera and LIDAR data. (2) "Anchor-fusion" occurs at the RPN stage, adequately fusing feature crops from different views guided by anchors to achieve a high recall at the next detection step. (3) "Proposal-fusion" occurs at the detection stage, applying innovative attention models as feature selectors to refine the fusion by enhancing desirable features and suppressing inefficient ones.

The proposed architecture encompasses the following contributions:

- We design a pre-fusion method that encodes the reflection intensities of LIDAR point clouds into an additional channel for camera images, which obtains a new image representation called RGB-Intensity (RGB-I). This aims to strengthen the representational power of multiple modalities using prior knowledge to enhance detection performance.

- We propose a novel proposal-fusion method that incorporates the self-attention mechanism into the network structure. This utilizes a newly designed module, called proposal-element attention (PEA), to adaptively re-weight elements of proposals from different views to determine how much the features of the proposals from each view contribute to the fusion and subsequent detection. This makes the fusion results include efficient and critical information.
- Inspired by the Cascade R-CNN [20] method for 2D object detection, we propose a cascade-enhanced detector consisting of a sequence of detectors trained with increasing intersection over union (IoU) thresholds. This makes the network more selective against close false positives, allowing for a higher 3D localization accuracy for small classes.

Our experimental evaluation on the KITTI benchmark suite [21] shows that our MCF3D method outperforms most state-of-the-art multi-sensor-based methods, and achieves a strong performance with a low computational cost and memory footprint.

II. RELATED WORK

A. LIDAR-BASED METHODS FOR 3D OBJECT DETECTION

1) USING RAW POINT CLOUD DATA

LIDAR scans object to generate unstructured point clouds represented by the 3D spatial coordinates and reflectivity (x, y, z, r) [22]. PointNet [8] and PointNet++ [23] directly digest raw point clouds to learn pointwise features for 3D object classification and semantic segmentation. Reference [8] employs a CNN-based architecture with t-net networks and symmetric functions to solve rotation and disorder problems. Reference [23] improves [8] that could learn local structures at different scale by means of set abstraction levels and density adaptive layers.

2) CONVERTING POINT CLOUDS TO OTHER FORMATS

VoxelNet [6] quantizes raw point clouds using voxel grids, and combines pointwise features with locally aggregated features through voxel feature encoding (VFE) layers. Then, either 2D or 3D convolutional networks are utilized on the voxelized shapes to detect 3D objects. However, the high computational cost of 3D convolutions constrains the inference speed and voxel resolution. SECOND [24] extends [6] further by applying sparse convolution to reduce the computational, and uses novel loss function to improve orientation regression performance. PIXOR [2] encodes point clouds as the BEV representation, and then using a fine-tuned RetinaNet [25] to achieve 3D real-time detection. Complex-YOLO [26] takes voxelwise BEV features extracted by MV3D [16] network as input, using a YOLO [27] network to detect 3D objects with a high frame-rate at approximately 50fps on an NVIDIA Titan Xp GPU.

These LIDAR-based methods have been shown to perform well for 3D object detection by only learning geometric spatial features from point cloud data. However, their capacities to detect small objects (such as pedestrians and cyclists) need to be improved. Furthermore, if they are extended to multi-tasking problems such as segmentation and re-recognition, LIDAR information alone is not sufficient.

B. CAMERA-BASED METHODS FOR 3D OBJECT DETECTION

1) STEREO-BASED

3DOP [10] estimates depth information from stereo images, and generates 3D box proposals by encoding the depth and hand-crafted geometric features into an energy minimization framework. 3D proposals are then fed into a modified Fast R-CNN [28] pipeline to predict the 3D object coordinates and classes. Stereo R-CNN [11] simultaneously detects and associates object in left and right images, exploiting dense object constraints in raw stereo images to formulate the projection relations for estimating and regressing final 3D bounding boxes.

2) MONOCULAR-BASED

Mono3D [9] takes advantage of object size priors, ground-plane priors, and monocular RGB images to score 3D proposals in an energy minimization approach. It has a similar framework to 3DOP [10], and produces more accurate 3D proposals than that method.

These camera-based methods do not perform well on 3D detection tasks compared with LIDAR-based methods, because the inaccurate depth information estimated from monocular or stereo images can seriously affect the learning and understanding of 3D spatial distributions for neural networks.

C. MULTI-SENSORS-BASED METHODS FOR 3D OBJECT DETECTION

Autonomous driving requires 3D object detection with a high accuracy and good stability, which cannot be satisfied by a single sensor. Therefore, many multimodal fusion methods utilizing both LIDAR and camera sensors have been proposed. These methods can be categorized into two classes. Those in the first class apply mature 2D object detectors to generate proposals from 2D images, then map these to point clouds for extrusion and regression. Those in the other propose 3D region proposal networks (RPN), which exploit BEV-only or the BEV with a camera image to generate 3D proposals for subsequent 3D object detection.

1) 2D-DRIVEN 3D OBJECT DETECTION

F-PointNet [12] selects necessary frustum-region reasoned point clouds by utilizing initial detection results generated by a 2D detector. Then, two cascaded PointNet [8] pipelines are consecutively applied to process the reduced point clouds to conduct 3D object instance segmentation and detection. RoarNet [13] is composed of RoarNet 2D and RoarNet 3D pipelines: the former one estimates proposals' 3D poses and 2d bounding boxes from 2D monocular images, then deriving more proposals locations based on geometrically feasible to reduce sensors out of synchronization problem. The later one is analogous to Faster R-CNN [29], taking these 3D region point clouds which have the shape of standing cylinders generated by the former one as inputs, then predicting 3D bounding box based on [8]. F-ConvNet [14] extends [12] that using multiple duplicate PointNet pipelines to respectively process a sequence of frustums generated for each region proposal and then applies the fully convolutional network to regress final 3D bounding boxes. However, the accuracy of these 2D-driven 3D approaches is bounded by the accuracy of the 2D detections, which is susceptible to snowy and low-light scenes.

2) 3D REGION PROPOSAL NETWORKS

MV3D [16] converts LiDAR point clouds into 2D BEV and front view (FV) representations by means of exploiting height, intensity, and density maps extracted from voxelized point clouds. Using a modified RPN of Faster R-CNN [29] to generate 3D proposals from the BEV feature map. Then, the proposals are projected onto BEV, FV, and 2D images feature maps to obtain the corresponding features, and these are fed into a deep fusion method to produce the final 3D detection results. Compared with the method in [16], AVOD [17] reforms and reduces the handcrafted inputs to speed up processing. Two modified VGG-16 [30] networks are utilized to generate high-resolution feature maps from images and the corresponding BEV, allowing small objects to be located more effectively. Then, these full resolution features, cropped by prior anchors, are fused to produce more reliable 3D object proposals for final 3D object detection. ContFuse [19] employs continuous convolutions to aggregate image and BEV features at different resolution levels based on their geometric position relationships. This approach utilizes dense image features to enrich sparse LIDAR point clouds to enhance the 3D detection performance.

Similar to 3D region proposal methods, we take advantage of a two-stage architecture for 3D object detection. However, our network incorporates several improvements and achieves a better performance.

III. MCF3D ARCHITECTURE

The main idea behind MCF3D is to construct an excellent data-driven network that makes full use of the display and implicit information of multimodal data at each stage, adding "knowledge guidance" and "complementary fusion" to accomplish efficient training and accurate predictions for 3D object detection. Fig. 1 illustrates the architecture of MCF3D, which consists of three components: (1) multimodal input representations; (2) a 3D RPN; and (3) second stage detector(s).

A. MULTIMODAL REPRESENTATIONS

We utilize prior knowledge and spatial geometric constraints to process 2D camera images and LIDAR point clouds, making input representations accurate and compact to maintain the computational speed while effectively extracting space, texture, and color information.

1) BIRD'S-EYE VIEW REPRESENTATION

Referring to VoxelNet [6], MV3D [16], and AVOD [17], we generate a six-channel BEV map encoded by the height and density to represent LIDAR point clouds. First, the raw LIDAR point clouds are cropped at $[40, 40] \times [0, 70]$ m. Point clouds that are projected outside of the image boundaries are removed, and those that remain are quantized with voxel grids at a resolution of 0.1 m. Then, the voxelized point clouds are divided into five equal slices between [0, 2.5] m along the Z-axis to obtain the first five channels of the BEV map. Each slice is encoded via the maximum height of the points in each grid cell. Finally, the sixth BEV channel is encoded via the point density information of each cell in all the voxelized point clouds, which is computed as $\min\left(1.0, \log(N+1) \big/ \log(64)\right)$, where N is the number of points in a cell.

2) RGB-I REPRESENTATION

To a certain extent, the reflection intensity information provided by LIDAR point clouds represents the materials of objects. For example, the reflection intensity of a road surface is between (0 – 0.1) and those of cars, people, and trees are between (0.1 – 0.8) in the KITTI dataset. Fig. 2(a) and Fig. 2(b) visualize point cloud representations with the reflection intensity and distance information, respectively. It can be observed that the expression in the form of the reflection intensity is closer to the human visual sense than the distance information. As shown in Fig. 2(a), cars (V0, V1) and cyclists (C0, C1) are rendered as corresponding colors according to categories. In addition, the intrinsic attributes of the reflection intensity and color are more consistent, and a combination of these two similar features will be more easily learned by the neural network. Moreover, the reflection intensity information not only reflects an object's spectral information, but

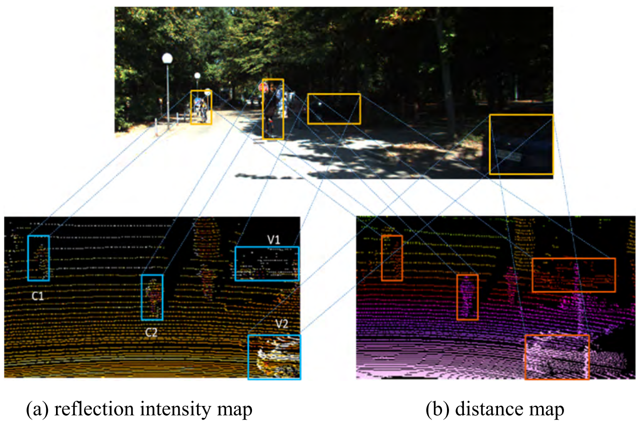

FIGURE 2. Illustration of relations for color, reflection intensity, and distance (Section 3.A.2). (a) shows the reflection intensity map visualized from point clouds corresponding to the images at the top. (b) shows the distance map visualized from the same point clouds as in (a).

is also not affected by extreme weather and lighting conditions and can effectively represent objects under a shadow to improve the detection performance, as shown in Fig. 2.

Therefore, we propose a novel pre-fusion method to convert RGB images into RGB-I representations. According to the spatial geometric relationship between the two sensors, the accurate but sparse reflection intensity information of LIDAR point clouds is innovatively added to the corresponding RGB image as an additional channel, which has not been performed in previous related networks, as visualized in Fig. 3. The detailed procedure is as follows: First, point clouds are transformed from the 3D LIDAR coordinate system to the camera coordinate system via extrinsic parameters, which represent the positional relationship between the two sensors to align the two modalities. Second, 3D point clouds are projected in the camera coordinates onto the 2D camera image. This process is formulated as in (1) and (2).

$$\begin{pmatrix} u \\ v \end{pmatrix} = P_{rect} \cdot T_{velo}^{cam} \begin{pmatrix} x \\ y \\ z \end{pmatrix} \tag{1}$$

$$T_{velo}^{cam} = \begin{pmatrix} R_{velo}^{cam} & t_{velo}^{cam} \\ 0 & 1 \end{pmatrix} \tag{2}$$

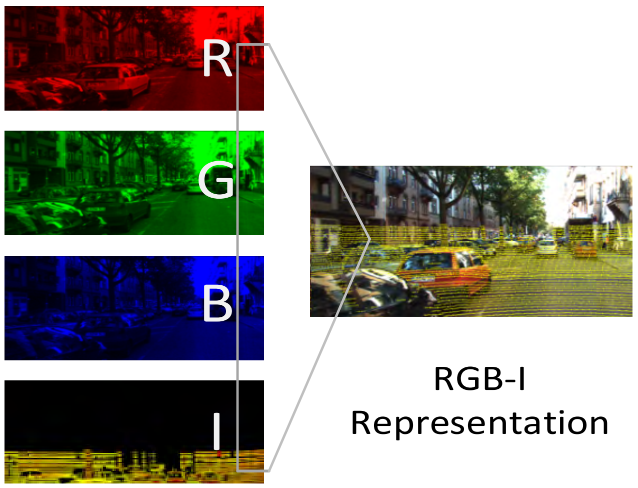

FIGURE 3. Illustration of RGB-I Transformation. The reflection intensities of lidar point clouds are encoded into an additional channel for camera images (section 3.a.2). "I" is the intensity map visualized from point clouds.

Following the KITTI benchmark suite [21], where (x, y, z) are 3D point coordinates in the LIDAR coordinate system, (u, v) are the camera pixel coordinates corresponding to (x, y, z). Furthermore, T_{velo}^{cam} is the rigid body transformation from the LIDAR coordinate system to the camera coordinate system, consisting of a rotation matrix R_{velo}^{cam} and translation vector t_{velo}^{cam}, as shown in (2). In addition, P_{rect} is a projection matrix that converts a 3D point in the camera coordinate system to a 2D point in the camera image. The calibration parameters mentioned above are also provided in [21].

Using the above procedure, we determined the corresponding position of each LIDAR point in the camera image. The pixel at this position consists of three channels, R, G, and B. Then, the reflection intensity of the corresponding LIDAR point is added as the fourth channel. If one pixel corresponds to more than one point, then the reflection intensities of the points are averaged. Because the resolution of the LIDAR point clouds is significantly lower than that of a camera image, only a small number of pixels (approximately 1/10) in images will obtain a reflection intensity, and the other pixels are given a value of 0 in the fourth channel. The RGB-I representation only makes use of the reflection intensity information of point clouds that is not utilized when generating the BEV representation in Section 3.A.1, adding little additional computational costs.

B. 3D REGION PROPOSAL NETWORK

Inspired by state-of-the-art 2D [20], [28], [29], [31] and 3D [16], [17] object detectors, we propose a 3D RPN, which is similar to a weak detector, to generate non-oriented region proposals with a high recall and low precision. First, we exploit the designed feature extractors to produce feature maps of BEV and images. Second, we generate 3D anchors and then project these onto the two view feature maps, respectively. Fusing the projected corresponding regions via the proposed anchor fusion method. Third, the fused features are fed to the proposed multi-layer perceptron (MPL) to regress the difference between the anchors and ground truth to generate 3D proposals.

1) THE FULL-RESOLUTION FEATURE EXTRACTOR

Inspired by [17], two identical CNNs are respectively applied onto the RGB-I and LIDAR BEV representations to extract features. Each CNNs consists of an encoder and a decoder. The encoder is a modified and simplified VGG-16 [30], with the number of channels reduced by half in each convolutional layer, the conv-5 layer and those below removed. Its output feature maps have high-level semantics but a low-resolution, resulting in small classes such as pedestrian and cyclist only retaining little information. Therefore, the design of the decoder is inspired by the feature pyramid network (FPN) [31] to up-sample the feature map generated by the encoder back to the input resolution. Then, feature maps of the same spatial size from encoder and decoder are merged via a 3×3 convolution and concatenation operation. The final outputs have a high resolution and strong semantics, and are shared by the 3D RPN and the second stage detector(s).

2) 3D ANCHOR GENERATION AND FUSION

The 3D Anchors are encoded by the centroid (c_X, c_Y, c_Z) and axis-aligned dimensions (d_X, d_Y, d_Z) [17]. The (c_X, c_Y) pairs are sampled at intervals of 0.5 m in the BEV, while c_z is a fixed value that is determined according to the sensor's height above the ground plane. Because we do not regress the orientation at this stage, (d_X, d_Y, d_Z) are transformed from (w, l, h) of the prior 3D boxes based on rotations. Furthermore, (w, l, h) are determined by clustering the ground truth object sizes for each class in the training set [16].

In [17], AVOD exploits 1×1 convolutions on the BEV and image feature maps, reducing their dimensionalities from 32 to 1 and allowing the RPN to process anchors with only a small memory overhead. The complete features are used to predict the final results at the second stage, while the simplified features are used to generate region proposals at the present stage. However, this approach changes the feature distribution, which increases the difficulty of network learning and leads to a decline in the final detection performance. We directly project each 3D anchor onto the original BEV and image feature maps to obtain two corresponding region-based features. Then, these are respectively cropped from two view feature maps and the cropped features are resized to be equally sized. These should be the same size as the proposals used for subsequent detection, unlike in [17]. Next, these fixed-length feature crops from both views are fused via a concatenation operation, which is more inclusive and comprehensive than the element-wise mean operation employed in [17].

3) 3D PROPOSAL GENERATION

These fused feature crops are fed to two similar branches to perform 3D proposal box regression and binary classification, respectively. Each branch consists of three convolution layers rather than fully connected layers. The regression branch estimates $(\Delta c_X, \Delta c_Y, \Delta c_Z, \Delta d_X, \Delta d_Y, \Delta d_Z)$, representing the differences between the centroids and dimensions of the anchors and target proposals. The classification branch predicts the "objectness" score, which is used to determine whether an anchor is an object or background. During training, the 3D anchors and ground truth bounding boxes are projected onto the BEV to compute the 2D IoUs between these, which are employed to determine positive, negative, and ignored anchors for training. The IoU thresholds are set to different values depending on the object class (introduced in Table 1 and Section 3.D). We optimize the following multitasking loss function in the 3D RPN stage.

$$L_{RPN} = \beta_1 L_{obj} + \beta_2 L_{loc+size} \tag{3}$$

Here, $L_{loc+size}$ is a smooth L1 loss for the 3D proposal box regression task, L_{obj} is a cross-entropy loss for the "objectness," $\beta_1 = 1.0$ and $\beta_2 = 5.0$ are constant coefficients of our RPN loss formula. 2D non-maximum suppression (NMS) is applied to remove redundant regressed anchors (3D proposals) based on the predicted "objectness" scores. With an IoU threshold of 0.8, the NMS keeps the top 1024 3D proposals for the subsequent detection stages.

C. SECOND STAGE DETECTION NETWORK

The proposed second stage detection network aims to further optimize and regress the above non-oriented region proposals to achieve a final excellent 3D object detection performance. First, we propose a novel attention-guided fusion method to effectively and adaptively combine paired proposals from multiple views. Second, the fused region features are fed to the specific branches for dimension refinement, orientation estimation, and category classification. In addition, we propose a cascade-enhanced detector modified from [20], which consists of three stages employing two detectors in succession to enhance the detection performance for small classes such as pedestrian and cyclist.

1) ATTENTION-GUIDED PROPOSAL FUSION METHOD

3D proposals are respectively projected onto the two view feature maps. These projected features are cropped and resized to be equally sized and taken as the inputs of the fusion operations. In current multi-sensor-based methods, the weights of the proposals (crops) from different views are fixed when fusing these, which inevitably affects the representation of critical information and limits a network's adaptivity to various appearances. For example, in Fig. 2 because the cars (V0, V1) are in shadow in the image, their detection should primarily apply features from the point clouds. The cyclist C0 is far away from the sensors and can only correspond to a few LIDAR points, and so detecting the cyclist should depend on the features from the image. Therefore, our goal is to allow the network to determine how much each view proposal contributes to the fusion and subsequent detection and let important and efficient features contribute more while inefficient ones contribute less.

Therefore, inspired by [32]–[36] we propose a simple but effective attention module named Proposal-Element Attention (PEA) as a multimodal feature selector to guide the proposal fusion. The PEA module can emphasize informative elements and suppress unnecessary ones by learning the input features themselves. In the self-attention mechanism, channel-wise [32] and spatial [36] attention focus respectively on "what" and "where." Our PEA approach absorbs and references the advantages of these, using fewer operations and parameters to achieve "element-wise" attention to focus on "which." Its structure is illustrated in Fig. 4. The PEA module consists of a simple encoding part and a decoding part: Given a proposal-features $P \in \mathbb{R}^{(C\times H\times W)}$ as input, P is first aggregated across its spatial dimensions $(H \times W)$ using a global average pooling operation, generating a channel descriptor $P_{avg} \in R^{(C\times 1\times 1)}$. Then, the encoded features P_{avg} are forwarded to the multi-layer perceptron (MLP) consisting of fully connected (FC) layers and activation functions. Unlike in [32]–[36], the FC layers increase the dimensions layer by layer. The first has an activation size set to $(C/r \times H \times W)$ followed by a ReLU [37]. The second has an activation size of $(C \times H \times W)$ followed by a sigmoid, where r is the reduction ratio. Next, a reshape operation is applied to convert the one-dimensional (1D) attention map into our element attention map $M \in R^{(C\times H\times W)}$, which has the same dimensions as the input P. Finally, an element-wise multiplication operation is applied to perform feature re-weighting. The whole element-wise attention process can be summarized as follows:

$$M(P) = \sigma(MLP(AvgPool(P))) = \sigma\left(W_1\left(\delta\left(W_0\left(P_{avg}\right)\right)\right)\right) \tag{4}$$

$$P' = M(P) \otimes P \tag{5}$$

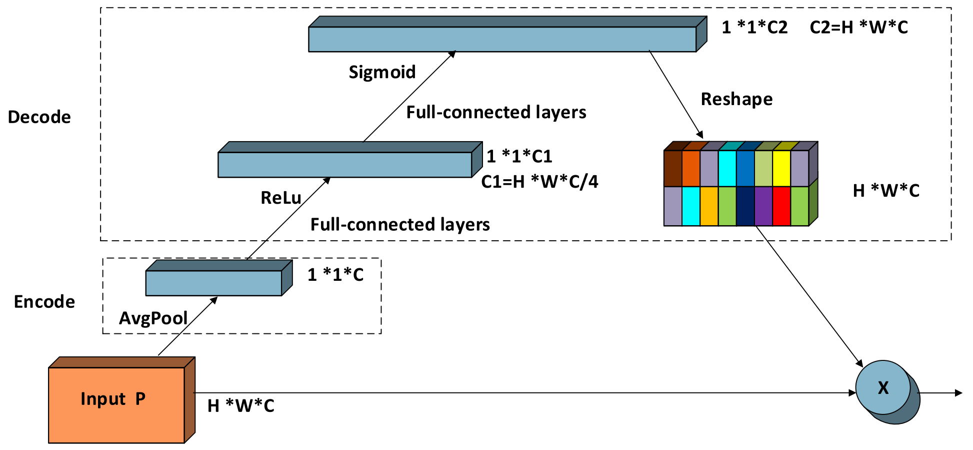

FIGURE 4. Diagram of the proposal-element attention (PEA) module (Section 3.C.1).

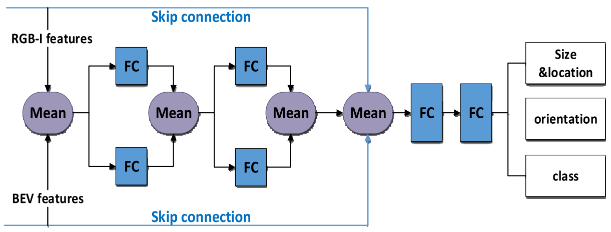

FIGURE 5. Architectures of the skip fusion method. "Skip connection" indicates that the inputs skip over the intermediate processing part directly to the output (Section 3.C.1). "Mean" denotes element-wise mean operation. "FC" stands for a fully-connected layer.

where σ and δ denote the sigmoid and ReLU activation functions, respectively; $W_0 \in R^{(C/r \times H \times W)}$ and $W_1 \in R^{(C \times H \times W)}$ refer to the MLP weights; $\otimes$ denotes the element-wise multiplication operation; P' is the final refined output. PEA is a lightweight module, which can be seamlessly integrated into our architecture and jointly trained with other parts of our MCF3D in an end-to-end fashion with a slight additional computational cost.

We employ two PEA modules to process the two respective view proposal features. Following the attention-guided reweighting procedure, both features are fed into our "skip fusion" approach, as shown in Fig. 5. Inspired by [16], [17], this approach enables interactions between intermediate layers from different views. First, we apply an element-wise mean operation to fuse both inputs since it is more flexible when combined with drop-path training, and separate FCs are utilized to learn the feature transformations independently. Then, the above operations are repeated once to obtain the intermediate outputs. Next, we innovatively apply "skip connections" to directly fuse the primary inputs and intermediate outputs. This makes the fusion results include clearly original and repeatedly abstracted features. Finally, these are passed into a series of FCs for the subsequent prediction.

2) GENERATING THE FINAL DETECTION GENERATION

The above fused feature crops are processed through three parallel fully connected layers for output box regression, orientation estimation, and category classification, respectively. Similar to [17], a 3D bounding box is simply encoded with a 10-dimensional vector, which reduces redundancy while keeping the physical constraints. We perform 3D bounding box regression by regressing to $(\Delta x_1 \ldots \Delta x_4, \Delta y_1 \ldots \Delta y_4, \Delta h_1, \Delta h_2)$ that represent the top and bottom corner offsets from the ground plane. The estimated orientation vector is represented as $(\cos(\theta), \sin(\theta))$, that implicitly handles angle wrapping. During training, we optimize the following multitasking loss function at the second stage:

$$L_{detection} = \beta_1 L_{cls} + \beta_2 L_{loc+size} + \beta_3 L_{ori} \tag{6}$$

where $L_{loc+size}$ and L_{ori} are the regression losses for the location and size and the orientation, respectively, which are represented as smooth L1 losses. Furthermore, L_{cls} is the classification loss, which is represented as a cross-entropy loss, and $\beta_1 = 1.0$, $\beta_2 = 5.0$, $\beta_3 = 1.0$ are constant coefficients of our detection loss formula. Whether the proposals are considered in the computation of the above loss depends on their 2D IoU in the BEV with the ground truth boxes, as introduced in Table 1 and Section 3.D. To remove overlapping and redundant detections, the 2D NMS is employed with an IoU threshold of 0.01 in the BEV.

3) CASCADE-ENHANCED DETECTOR

Inspired by [20], we cascade an extra detector to the above network to enhance the 3D object detection performance for small classes, with the resulting design called MCF3D(cascade). The network architecture is also changed from two stages to three stages, as shown in Fig 6. The added detector (v2) has a similar structure to the former detector (v1) described in Section 3.C.2. To control the number of parameters, the detector (v1) is changed to utilize early fusion [16] and (v2) uses our skip fusion method (introduced in Section 3.C.1). The two detectors are trained using different and increasingly higher IoU thresholds, with the aim of detecting true positives while suppressing close false positives. The output of the first detector is taken as the input to the second one, which is equivalent to obtaining an

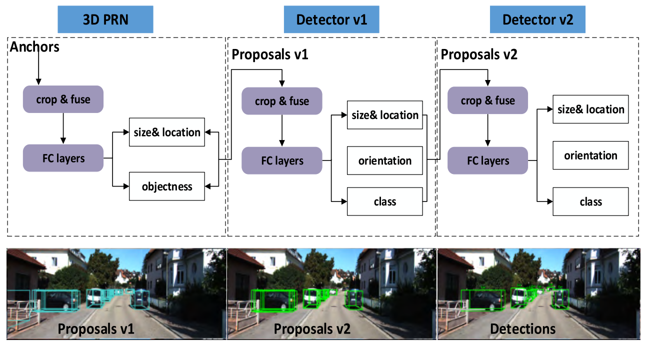

FIGURE 6. Top row: a simple detection pipeline of MCF3D (cascade). Bottom row: Visualizations of outputs of the corresponding top parts (Section 3.C.3). "FC layers" stands for fully-connected layers.

increasingly good distribution stage by stage to train a higher quality detector. Cascade regression is exploited as a resampling mechanism, to keep the set of positive examples for the successive stages at a roughly constant size. When faced with the detection of small classes, the cascade detectors adapted to increasingly higher IoUs can be effectively trained while overcoming the overfitting problem, and the same cascade procedure is applied at the inference stage. The performance of the cascaded detectors is highly dependent on the design of the IoU thresholds required to define positives/negatives at each stage, as described in Table 1 and further discussed in Sections 3.D and Section 4.B.3. The three-stage network including the extra detector can be trained in an end-to-end or multi-step manner by optimizing the following loss function:

$$L_{\text{total}} = \left(L_{\text{RPN-obj}} + L_{\text{RPN-reg}}\right) + \left(L_{\text{v1-cls}} + L_{\text{v1-reg}}\right) + \left(L_{\text{v2-cls}} + L_{\text{v2-reg}}\right) \quad (7)$$

where $L_{\text{RPN-obj}}$, $L_{\text{v1-cls}}$, and $L_{\text{v2-cls}}$ are the cross-entropy losses for classification in the RPN and two detection networks, and $L_{\text{RPN-reg}}$, $L_{\text{v1-reg}}$, and $L_{\text{v2-reg}}$ are the L1 losses for the non-oriented and oriented bounding box regressions.

D. TRAINING DETAILS

We train two network versions: MCF3D and MCF3D (cascade). We also train two separate networks for each version, for car and pedestrian/cyclist respectively. MCF3D consists of an RPN and a detector, and the two stages are jointly trained in an end-to-end fashion using mini-batches containing one image with 512 and 1024 ROIs, respectively. Approximately 50% of the ROIs are kept as positive for each mini-batch. Whether the anchors/proposals are considered as positive, negative, or ignored samples for training relies on the IoUs between them and all the ground truth boxes. Detailed descriptions of each stage's IoU thresholds are given in Table 1. For the car class in MCF3D, an anchor/proposal is considered as positive if its maximum IoU is above 0.5/0.65 (in the BEV), and is treated as negative if its maximum IoU is below 0.3/0.55. For the pedestrian and cyclist classes in MCF3D, an anchor/proposal is considered as positive if its maximum IoU is above 0.45/0.55, and as negative if its maximum IoU is below 0.3/0.45. The network is trained using stochastic gradient descent (SGD) for 150K iterations without a pre-trained model. The Adam optimizer is utilized with an initial learning rate of 0.0001, which is decayed exponentially every 100K iterations with a decay factor of 0.1. Training the MCF3D network using a single Titan Xp GPU took 17 h.

MCF3D (cascade) includes an RPN, detector (v1), and detector (v2). The three stages are trained in a two-step fashion using mini-batches containing one image with 512, 1024, and 1024 ROIs, respectively, and approximately 50% of the ROIs are kept as positive for each mini-batch. The negative/positive IOU thresholds for each stage for the three classes are introduced in Table 1. In the first step, we fix the parameters of the detector (v2) to the initialized values, and only update the parameters of the RPN and the detector (v1). In the second step, we lock the parameters of the RPN and detector (v1) to the results trained in the first step, and only update parameters of the detector (v2). The above two training stages both adopt the following settings: SGD is utilized with the Adam optimizer for 120K iterations, with an initial learning rate of 0.0001 and an exponential decay factor of 0.1, decaying every 100K iterations. Training the MCF3D(cascade) network using a single Titan Xp GPU took 34 h.

TABLE 1. IoU threshold settings for each stage of MCF3D and MCF3D (cascade).

The Stage	IoU Threshold	MCF3D		MCF3D (cascade)	
		Car	Ped. & Cyc.	Car	Ped. & Cyc.
RPN	Negative IoU thresholds	[0,0.3]	[0,0.3]	[0,0.3]	[0,0.3]
(First stage)	Positive IoU thresholds	[0.5,1]	[0.45,1]	[0.5,1]	[0.4,1]
Detector v1	Negative IoU thresholds	[0, 0.55]	[0,0.45]	[0,0.45]	[0,0.35]
(second stage)	Positive IoU thresholds	[0.65,1]	[0.55,1]	[0.55,1]	[0.45,1]
Detector v2	Negative IoU thresholds	-	-	[0,0.55]	[0,0.45]
(third stage)	Positive IoU thresholds	-	-	[0.65,1]	[0.55,1]

IV. EXPERIMENTS

We evaluate our MCF3D's performance on the KITTI object detection benchmark [21] for 3D object detection and BEV object detection (3D localization) tasks on the car, pedestrian, and cyclist classes. Following the KITTI settings, each class is evaluated based on the three difficulty levels of easy, moderate, and hard considering the object size, distance, occlusion, and truncation. The KITTI dataset provides 7,481 images/point clouds for training and 7,518 for testing. Because the ground truth for the test set is not available and access to the test server is limited, we conduct a comprehensive evaluation using the protocol described in [9], [10], [16], and [17]. The 7481 provided training frames are

TABLE 2. Comparison of the 3D Object and localization performance of MCF3D with state-of-the-art 3D object detectors.

Class	Method	Runtime(s)	Easy	Moderate	Hard	Easy	Moderate	Hard
Car	AVOD-FPN [6]	0.1	84.41	74.44	68.65	-	-	-
	MV3D [5]	0.36	71.29	62.68	56.56	86.55	78.10	76.67
	F-PointNet [18]	0.17	83.76	70.92	63.65	88.16	84.02	76.44
	VoxelNet [12]	0.23	81.97	65.46	62.85	89.60	84.81	78.57
	ContFusion [23]	**0.06**	**86.32**	73.25	67.81	**95.44**	**87.34**	82.43
	Ours	0.14	84.36	**76.09**	**75.13**	89.41	86.98	**86.89**
	Ours (cascade)	0.16	84.11	75.19	74.23	88.82	86.11	79.31
Ped.	AVOD-FPN [6]	**0.1**	-	58.8	-	-	-	-
	F-PointNet [18]	0.17	70.00	61.32	53.59	72.38	**66.39**	59.57
	VoxelNet [12]	0.23	57.86	53.42	48.87	65.95	61.05	56.98
	Ours	0.13	68.54	64.93	59.47	68.56	64.98	59.55
	Ours (cascade)	0.15	**72.67**	**65.92**	**63.17**	**72.68**	66.06	**63.20**
Cyc.	AVOD-FPN [6]	**0.1**	-	49.7	-	-	-	-
	F-PointNet [18]	0.17	77.15	**56.49**	**53.37**	81.82	**60.03**	**56.32**
	VoxelNet [12]	0.23	67.17	47.65	45.11	74.41	52.18	50.49
	Ours	0.13	78.18	51.06	50.43	78.18	51.09	50.45
	Ours(cascade)	0.15	**82.69**	55.33	49.11	**82.69**	55.33	49.11

Average Precision (AP-3D) (in %) of 3D boxes and Average Precision (AP-BEV) (in %) of bird's eye view boxes on KITTI's validation set. We use a 3D IoU threshold of 0.7 for the Car class, and 0.5 for the pedestrian and cyclist classes.

split into a training and validation set at approximately a 1:1 ratio to avoid the same samples being included in both the training and validation set. We follow the official KITTI evaluation protocol, evaluating both 3D object detection and BEV object detection (3D localization) tasks at a 0.7 IoU threshold for the car class and 0.5 IoU threshold for the pedestrian and cyclist classes. The evaluation results are given in terms of the average precision (AP) metric. The runtimes refer to the inference times of our networks for one image on a Titan Xp GPU. The experiments are divided into three parts. First, we compare with state-of-the-art methods for 3D object detection on KITTI. Second, we perform some ablation studies. Third, we present qualitative results and discuss the strengths and limitations of our methods.

A. COMPARING WITH STATE-OF-THE-ART METHODS

We evaluate two versions of our implementation (introduced in Section 3.D). Ours refers to MCF3D containing an RPN and a detector, and Ours(cascade) refers to MCF3D(cascade) containing an RPN and the two detectors (v1, v2). Table 2 presents the results of our methods on the KITTI validation set for 3D object detection and localization (BEV) tasks. Our method outperforms several state-of-the-art algorithms, including LIDAR-based [6] and multi-modal-based [12], [16], [17], [19] methods. F-PointNet [12] employs a 2D detector that has been fine-tuned using ImageNet [38] weights and MV3D [16] utilizes a pre-trained model for initialization, whereas our network is trained from scratch.

On the car class, Ours achieves a 1.65% increase on the most important "Moderate" column in 3D object detection compared with the second-best performing method, with noticeable margins of 6.48% and 4.46% for hard (highly occluded or far) instances in 3D and BEV object detection, respectively, indicating that our method is particularly effective on cluttered objects. On the pedestrian class, even with the only detector (v1), Ours performs reasonably effectively. Moreover, once we add the cascade detector (v2), Ours(cascade) ranks first in both tasks except for a slightly lower score than F-PointNet for moderate instances in BEV. In particular, for the hard difficulty Ours(cascade) achieves 9.58% and 3.63% increases in 3D and BEV, respectively, demonstrating that our cascade version is effective in detecting dense objects such as a crowd of people. On the cyclist class, our method only outperforms the state-of-the-art methods slightly on easy instances. We believe that this is because of the low number of cyclist instances in the KITTI dataset, which induces a bias towards pedestrian detection in our joint pedestrian/cyclist network. Concerning the inference times of our networks for one image on a Titan Xp GPU shown in Table 2, our runtime is slightly longer than those of the state-of-the-art methods, but it still achieves a comparable speed.

Overall, Ours can produce high-accuracy results on the car class with a fast inference speed, and Ours(cascade) performs better on small class predictions. The outputs of our networks are visualized in Fig. 8, where we observe accurate 3D box

TABLE 3. Effect of input representations.

Input (pre-fusion)	Car			Pedestrian			Cyclist		
	Easy	Mode	Hard	Easy	Mode	Hard	Easy	Mode	Hard
RGB+BEV	84.33	75.61	68.89	67.78	64.09	58.91	66.73	48.06	41.29
RGB-Distance + BEV	84.34	75.63	70.01	68.05	64.42	59.37	67.82	50.06	43.72
RGB-Density + BEV	83.89	74.60	69.17	67.75	63.97	59.38	66.80	49.13	42.63
RGB-Intensity+ BEV	**84.36**	**76.09**	**75.13**	**68.54**	**64.93**	**59.47**	**78.18**	**51.06**	**50.43**

(See Section 3.A.2 and Section 4.B.1) 3D object detection performances of detectors using different input representations. Average Precision (in %) of 3D boxes on KITTI's validation set. We use a 3D IoU threshold of 0.7 for the Car class, and 0.5 for the Pedestrian and Cyclist classes. The base detector is our MCF3D.

prediction even under very challenging cases. We also present several imperfect results in Fig. 9. We defer further discussion of success and failure case patterns to Section 4.C.

B. ABLATION STUDIES

In this section, we provide an analysis and ablation experiments to validate the components and variants of our proposed MCF3D on the KITTI validation set.

1) EFFECT OF INPUT REPRESENTATIONS

MV3D [16] shows that as input the RGB images contribute less to 3D object detection. Therefore, we convert this format to RGB-I (see Section 3.A.2) to strengthen the representational power through pre-fusion. In this section, we evaluate the performances when using different transformed representations, such as RGB-Distance and RGB-Density. The processes of generating these are similar to RGB-I, adding distance or density information, respectively, from point clouds to RGB images based on the spatial geometric relationships between the two sensors. "Density" indicates the number of points of point clouds that are transformed and projected onto the corresponding pixel position. Inspired by [16], this is computed as $\min(1.0, \log(N+1)/\log(64))$ for normalization, where N is the number of points.

In Table 3, we compare RGB-I, RGB-Density, and RGB-Distance with the original RGB representation, taking each of them respectively as the image input of our MCF3D method for 3D object detection. We observe that converting RBG to RGB-Density or RGB-Distance contributes negligibly to the performance. However, using RGB-I leads to a clear performance enhancement, especially at in the hard column, with gains of 6.24%, 0.56%, and 9.14% AP on the car, pedestrian, and cyclist classes, respectively. This phenomenon can be attributed to the closely related physical properties of the RGB and intensity, as described in Section 3.A.2. Fig. 7 presents a visualization of the results for using RGB and RGB-I in comparison to KITTI's ground truth. It can be observed that RGB-I could accurately detect some distant or heavily occluded cars that are not detected using RGB (e.g., V5, V6 in Fig. 7(c)), although these difficult cases are not labeled in KITTI (e.g., V3, V6 Fig. 7(b)).

2) EFFECT OF THE ATTENTION MODULE

To thoroughly evaluate the effectiveness of our proposed PEA (see Section 3.C.1), we take MCF3D without any attention module as the basic network, comparing its 3D detection results with those of the SE-enhanced [32], CBAM-enhanced [33] and PEA-enhanced ones. The three compared modules all have a self-attention mechanism. SE [32] exploits channel-wise attention to focus on "what," CBAM [33] exploits spatial and channel-wise attention to focus on both "what" and "where," and our PEA models the "element-wise" attention, using proposal-dependent features themselves to focus on "which." Table 4 shows that integrating our lightweight PEA into the network at the reduction ratio of four achieves the best result. This can be attributed to the fact that adaptively adjusting how much 'which' features in 'which' views contribute during fusion could suppress inefficient and interference information. To verify this assertion, Fig. 7 visualizes the output of MCF3D and MCF3D (no-attention) for comparison. The no-attention network fails to accurately exclude cluttered objects with similar shapes or colors with positive objects (e.g., V6 Fig. 7(d)) and is inefficient facing farther away objects (e.g., V7 Fig. 7(d)).

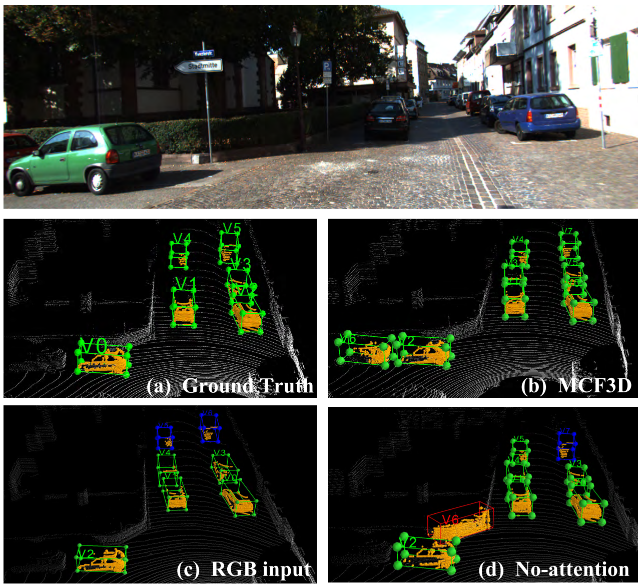

FIGURE 7. A qualitative comparison of our MCF3D, MCF3D with RGB inputs, and MCF3D without the attention module relative to KITTI's ground-truth on a sample from the validation set (best viewed in color with zoom in). True positive detection boxes are in green, while false positive boxes are in red and ground-truth boxes in blue are shown for false positive and false negative cases. Digit and letter beside each box denote instance ID and the class, with "V" for cars, "P" for pedestrian and "C" for cyclist. The orientation (driving direction) of each box is shown by the position of the digit and letter.

TABLE 4. Effect of attention module.

Attention Module		Car			Pedestrian			Cyclist		
		Easy	Mode	Hard	Easy	Mode	Hard	Easy	Mode	Hard
Base (no-attention)		83.85	74.18	68.34	66.71	64.08	57.82	77.44	50.01	48.71
Base + SE	r=4	84.02	74.83	68.58	66.89	64.13	57.91	78.01	49.86	49.33
Base + CBAM	r=8,k=7	83.86	74.56	68.19	65.31	63.22	56.75	75.92	48.56	48.63
	r=4,k=3	84.36	75.26	68.50	67.91	64.05	57.97	77.63	50.05	48.95
Base +PEA(ours)	r=8	**84.77**	76.03	69.05	67.54	63.68	58.96	77.83	49.88	49.75
	r=4	84.36	**76.09**	**75.13**	**68.54**	**64.93**	**59.47**	**78.18**	**51.06**	**50.43**

(See Section 3.C.1 and Section 4.B.2) 3D object detection performances of detectors using different attention module. Average Precision (in %) of 3D boxes on KITTI's validation set. We use a 3D IoU threshold of 0.7 for the Car class, and 0.5 for the Pedestrian and Cyclist classes. The baseline detector is our MCF3D without any attention module. "r" denotes the reduction ratio. "k" denotes kernel size in a convolution operation.

TABLE 5. Effect of the number of stages in MCF3D.

Architecture		Runtime (s)	Pedestrian			Cyclist		
			Easy	Mode	Hard	Easy	Mode	Hard
MCF3D	2-satge: RPN+Detector	0.13	68.54	64.93	59.47	78.18	51.06	50.43
MCF3D (cascade)	3-satge: RPN+2 Detectors	0.15	72.67	**65.92**	**63.17**	**82.69**	55.33	49.11
	4-stage: RPN+3 Detectors	0.17	**74.16**	65.23	63.16	82.67	**56.19**	**50.85**

(See Section 3.C.3 and Section 4.B.3) 3D object detection performances evaluated by Average Precision (AP3D) (in %) of 3D boxes s on KITTI's validation set. We use a 3D IoU threshold of 0.5 for the Pedestrian and Cyclist classes.

3) EFFECT OF CASCADE-ENHANCED DETECTORS

In Table 5, we evaluate the impact of the number of stages in our architecture on the 3D object detection performance. Owing to MCF3D(cascade) being designed with the aim of dealing with a lack of true positives (see Section 3.C.3), we only conduct experiments on the pedestrian and cyclist categories, which have fewer training examples than the car category. In the last row of Table 5, we present the results for with the four-stage version of MCF3D(cascade) that contains three detectors with a similar architecture. The method of adding the third detector is analogous to that in three-stage MCF3D(cascade), as described in Section 3.C.3. The results demonstrate that the three-stage version (in the second row) with an added detector (v2) results in a non-trivial improvement, adding a further detector (resulting in the four-stage version in the last row) only produces a slight increase or decrease. Considering that the number of MCF3D(cascade) parameters increases with the number of cascade stages, three-stage MCF3D achieves the best trade-off.

4) OTHER EFFECTS

Table 6 shows the effects of varying different hyper-parameters on the performance of our 3D object detector. In this section, we only focus on the car category, which has the most training examples and uses our MCF3D (in Section 3.D) as the base network. For the second row of Table 6, we utilize a shared feature extractor instead of two the same ones (as introduced in Section 3.B.1) to learn the features from both the point clouds and images. Although the number of parameters is reduced, this achieves considerably worse results than our base network. For the third row of

TABLE 6. Comparison of the performances for different variations of hyper-parameters.

Architecture	Runtime (s)	car		
		Easy	Mode	Hard
Base：MCF3D	0.14	84.36	**76.09**	**75.13**
Reuse feature extractor	0.13	76.76	67.20	59.38
RPN feature maps: 1×1 convolution [6]	**0.12**	84.03	75.25	68.40
PRN crops：3*3 ; proposal crops：7*7 [6]	0.13	**84.98**	75.38	68.84
PRN crops：5*5 ; proposal crops：5*5	0.13	84.43	75.74	74.86
Anchor-fusion: element-wise mean [6]	0.13	84.43	74.83	68.62
Proposal-fusion: deep fusion [5]	0.14	83.21	75.05	68.28

(See Section 4.B.4) Average Precision (in %) of 3D boxes on KITTI's validation set. We use a 3D IoU threshold of 0.7 for the Car class.

Table 6, referring to AVOD [17], we add 1×1 convolutional kernels to the respective outputs of the two feature extractors to reduce the memory overhead to compute anchor-specific feature crops. The modified model obtains inferior results to our base network. This can be attributed to the fact that when regressing proposals and predicting 3D boxes using different feature maps, it is difficult for the network to learn the inter-relationships in the feature distributions of different stages. For the fourth row of Table 6, we resize the anchors to 3×3 to generate proposals and resize the proposals to 7×7 for the final 3D detection regression process [17]. For the fifth row, both anchors and proposals are resized to 5×5. The parameters saved by these two methods are almost the same. However, the fifth-row module obtains significantly better results than the fourth-row module, which indicates

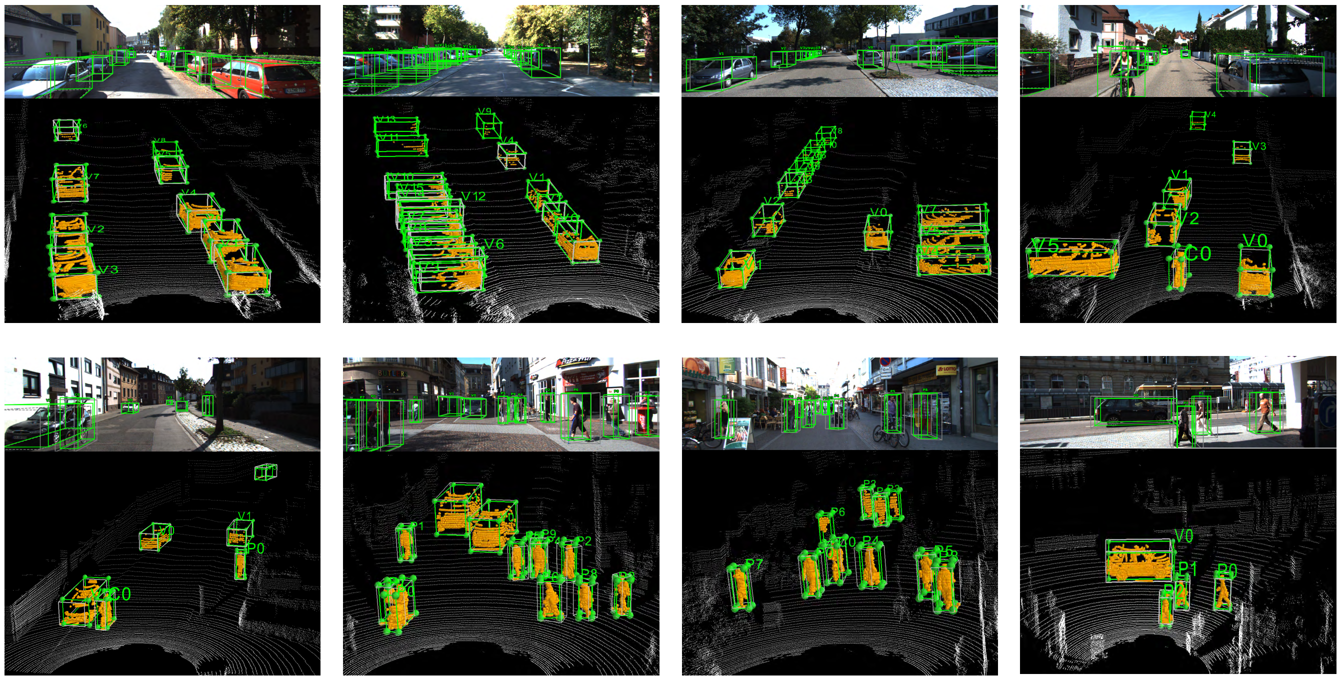

FIGURE 8. Visualizations of high-quality MCF3D results on the KITTI validation set (best viewed in color with zoom in). For each sample, the upper part is the image, the lower part is a representative view of the corresponding point cloud and points of each instance marked by yellow. The detection boxes are in green, while the ground-truth boxes are in gray. Digit and letter beside each box denote instance ID and the class, with "V" for cars, "P" for pedestrian and "C" for cyclist. The orientation (driving direction) of each box is shown by the position of the digit and letter. See Section 4.C for more discussion on the results.

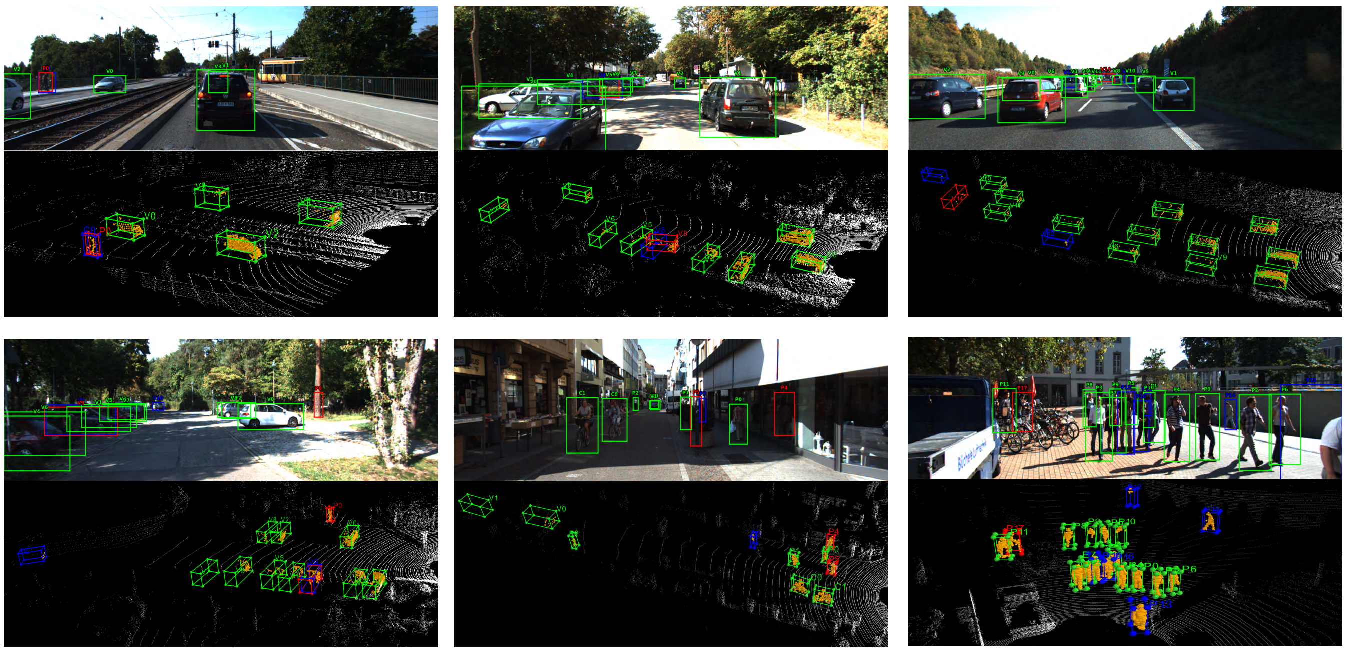

FIGURE 9. Visualizations of MCF3D failure results on the KITTI validation set (best viewed in color with zoom in). True positive detection boxes are in green, while false positive boxes are in red and ground-truth boxes in blue are shown for false positive and false negative cases. Digit and letter beside each box denote instance ID and the class, with "V" for cars, "P" for pedestrian and "C" for cyclist. The orientation (driving direction) of each box is shown by the position of the digit and letter. See Section 4.C for more discussion on the results.

that the RPN, as a weak detector, should have similar parameter settings to the second detector. For the sixth row of Table 6, we utilize an element-wise mean operation proposed by AVOD [17] instead of the concatenation operation for anchor fusion, which produces slightly inferior results to our base network. This verifies our assertion that rich features should be preserved as completely as possible in the early stages, even if some parameters are added. For the seventh row of Table 6, we apply the deep fusion method proposed by MV3D [16] for proposal fusion, rather than using our skip-fusion method (in Section 3.C.1). The results demonstrate that the skip connection further helps to process multimodal information in 3D object detection.

C. QUALITATIVE RESULTS AND DISCUSSION

In Fig. 8 we depict representative and high-quality outputs of our MCF3D. For easy and moderate cases that are fully visible or partly occluded at a reasonable distance, our network can produce particularly accurate oriented 3D bounding boxes. For certain challenging cases that are strongly overlapping or have a small number of available points, our model can predict correctly posed amodal 3D boxes (e.g., parallel parked cars in the first three columns of the first row and the crowd in the second column of the second row). Surprisingly, our network still successfully detects some difficult and complex cases that are heavily occluded or difficult to see, despite some not being labeled at all in the ground truth annotations (e.g., V8, V10, and V6 in the third column of the first row, and P3 and P6 in the third column of the second row).

In Fig. 9 we depict some failure detection results, which point towards the directions for improving our MCF3D approach. The first common mistake involves inaccurate rotations and size estimations owing to objects only have few points (e.g., V8 in the second column of the first row and V9 in the first column of the second row). The second type of failure includes false negatives on occluded or distant objects (e.g., V12 and V13 in the third column of the first row, V10 in the first column of the second row, and pedestrians in the last two columns of the second row). In addition, the detection of small classes is more challenging, and leads to some interesting failure modes. Pedestrians and cyclists are commonly misclassified as each other (e.g., C0 in the first column of the first row). Pedestrians are easily confused with narrow vertical features of the environment, such as poles or tree trunks (e.g., P0 in the first column of the second row and P4 and P5 in the second column of the second row).

V. CONCLUSIONS

In this study, we propose a sensory-fusion framework called MCF3D, consisting of a 3D RPN and a second-stage detector(s) subnet. The first stage produces non-oriented region proposals by fusing anchor-dependent features from LIDAR and camera views. The second stage performs dimension refinement, orientation estimation, and category classification by fusing proposal-dependent features from both views. We design different fusion methods for each stage according to the corresponding tasks. We innovatively introduce attention models (PEA) and a conversion input representation (RGB-I) to enhance the fusion effect and detection performance. Our experiments show that MCF3D outperforms state-of-the-art 3D detection methods by a large margin, especially for high-occlusion or crowded scenes. In the future, we will exploit an enhanced fusion method to reduce the gap between images and LiDAR and extend our network to achieve segmentation tasks.

REFERENCES

[1] W. Luo, B. Yang, and R. Urtasun, “Fast and furious: Real time end-to-end 3D detection, tracking and motion forecasting with a single convolutional net,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3569–3577.

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

[3] B. Li, “3D fully convolutional network for vehicle detection in point cloud,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2017, pp. 1513–1518.

[4] S.-L. Yu, T. Westfechtel, R. Hamada, K. Ohno, and S. Tadokoro, “Vehicle detection and localization on bird’s eye view elevation images using convolutional neural network,” in *Proc. IEEE Int. Symp. Saf., Secur. Rescue Robot. (SSRR)*, Oct. 2017, pp. 102–109.

[5] B. Li, T. Zhang, and T. Xia, “Vehicle detection from 3D lidar using fully convolutional network,” 2016, *arXiv:1608.07916*. [Online]. Available: https://arxiv.org/abs/1608.07916

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

[7] M. Engelcke, D. Rao, D. Z. Wang, C. H. Tong, and I. Posner, “Vote3Deep: Fast object detection in 3D point clouds using efficient convolutional neural networks,” in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May/Jun. 2017, pp. 1355–1361.

[8] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, “PointNet: Deep learning on point sets for 3D classification and segmentation,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 652–660.

[9] X. Chen, K. Kundu, Z. Zhang, H. Ma, S. Fidler, and R. Urtasun, “Monocular 3D object detection for autonomous driving,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 2147–2156.

[10] X. Z. Chen, K. Kundu, Y. Zhu, S. Fidle, R. Urtasun, and H. Ma, “3D object proposals using stereo imagery for accurate object class detection,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 5, pp. 1259–1272, May 2018.

[11] P. Li, X. Chen, and S. Shen, “Stereo R-CNN based 3D object detection for autonomous driving,” 2019, *arXiv:1902.09738*. [Online]. Available: https://arxiv.org/abs/1902.09738

[12] 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 Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 918–927.

[13] K. Shin, Y. P. Kwon, and M. Tomizuka, “RoarNet: A robust 3D object detection based on region approximation refinement,” 2018, *arXiv:1811.03818*. [Online]. Available: https://arxiv.org/abs/1811.03818

[14] 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: https://arxiv.org/abs/1903.01864

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

[16] 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.*, Jul. 2017, pp. 1907–1915.

[17] 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.

[18] Z. Wang, W. Zhan, and M. Tomizuka, “Fusing bird’s eye view LIDAR point cloud and front view camera image for 3D object detection,” in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2018, pp. 1–6.

[19] 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.

[20] Z. Cai and N. Vasconcelos, “Cascade R-CNN: Delving into high quality object detection,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, vol. 2018, pp. 6154–6162.

[21] A. Geiger, P. Lenz, and R. Urtasun, “Are we ready for autonomous driving? The KITTI vision benchmark suite,” presented at the IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2012. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/6248074

[22] Z. Li, Q. Dai, M. Q. Mehmood, G. Hu, B. Luk'yanchuk, J. Tao, C. Hao, I. Kim, H. Jeong, G. Zheng, S. Yu, A. Alù, J. Rho, and C.-W. Qiu, "Full-space cloud of random points with a scrambling metasurface," *Light, Sci. Appl.*, vol. 7, no. 1, Sep. 2018, Art. no. 63.
[23] 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.
[24] Y. Yan, Y. Mao, and B. Li, "SECOND: Sparsely embedded convolutional detection," *Sensors*, vol. 18, no. 10, p. 3337, 2018.
[25] 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.*, Oct. 2017, pp. 2980–2988.
[26] 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. (ECCV)*, Munich, Germany, Sep. 2018, pp. 197–209.
[27] 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.*, Jun. 2016, pp. 779–788.
[28] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 1440–1448.
[29] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.
[30] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: https://arxiv.org/abs/1409.1556
[31] 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. 936–944.
[32] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7132–7141.
[33] S. Woo, J. Park, J.-Y. Lee, and I. So Kweon, "CBAM: Convolutional block attention module," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 3–19.
[34] F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang, "Residual attention network for image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 3156–3164.
[35] S. Zhang, J. Yang, and B. Schiele, "Occluded pedestrian detection through guided attention in CNNs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6995–7003.
[36] L. Chen, H. Zhang, J. Xiao, L. Nie, J. Shao, W. Liu, and T.-S. Chua, "SCA-CNN: Spatial and channel-wise attention in convolutional networks for image captioning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 5659–5667.
[37] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. 27th Int. Conf. Mach. Learn. (ICML)*, 2010, pp. 807–814.
[38] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

JIARONG WANG was born in Changchun, Jilin, China, in 1989. She received the B.S. degree in optical engineering from the Changchun University of Science and Technology and the M.S. degree in circuits and systems from Jilin University. She is currently pursuing the Ph.D. degree with the Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, China. Her research interests include 2D and 3D object detections and stereo vision.

MING ZHU is currently a Research Fellow and a Supervisor of Ph.D. candidates of the Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences. His research interests include digital image processing, television tracking, and automatic target recognition technology.

DEYAO SUN received the B.E. degree in testing instrument and technology from the Harbin Institute of Technology, in 2014. He is currently pursuing the Ph.D. degree in mechatronics engineering with the Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, China. His research interests include object detection in remote sensing images and 3-Dimension images, scene understanding, and semantic segmentation.

BO WANG received the B.S. degree in microelectronics from Jilin University, in 2016. He is currently pursuing the Ph.D. degree with the Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, China. His research interests include object detection and 3D object detection.

WEN GAO is currently an Associate Research Fellow with the Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences. Her research interests include digital image processing, television tracking, and automatic target recognition technology.

HUA WEI received the B.S. degree in automation from Shandong University. She is currently pursuing the Ph.D. degree with the Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, China. Her research interests include object detection and fine-grained recognition.