IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 29, 2021, accepted October 11, 2021, date of publication October 14, 2021, date of current version October 26, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3120261

3D Object Detection With Multi-Frame RGB-Lidar Feature Alignment

EMEÇ ERÇELIK[1], (Member, IEEE), EKIM YURTSEVER[2], (Member, IEEE), AND ALOIS KNOLL[1], (Senior Member, IEEE)

[1]Chair of Robotics, Artificial Intelligence and Real-Time Systems, Technical University of Munich, 85748 München, Germany
[2]College of Engineering, Center for Automotive Research, The Ohio State University, Columbus, OH 43212, USA

Corresponding author: Emeç Erçelik (emec.ercelik@tum.de)

ABSTRACT Single-frame 3D detection is a well-studied vision problem with dedicated benchmarks and a large body of work. This knowledge has translated to a myriad of real-world applications. However, frame-by-frame detection suffers from inconsistencies between independent frames, such as flickering bounding box shape and occasional misdetections. Safety-critical applications may not tolerate these inconsistencies. For example, automated driving systems require robust and temporally consistent detection output for planning. A vehicle's 3D bounding box shape should not change dramatically across independent frames. Against this backdrop, we propose a multi-frame RGB-Lidar feature alignment strategy to refine and increase the temporal consistency of 3D detection outputs. Our main contribution is aligning and aggregating object-level features using multiple past frames to improve 3D detection quality in the inference frame. First, a Frustum PointNet architecture extracts a frustum-cropped point cloud using RGB and lidar data for each object frame-by-frame. After tracking, multi-frame frustum features of unique objects are fused through a Gated Recurrent Unit (GRU) to obtain a refined 3D box shape and orientation. The proposed method improves 3D detection performance on the KITTI tracking dataset by more than 4% for all classes compared to the vanilla Frustum PointNet baseline. We also conducted extensive ablation studies to show the efficacy of our hyperparameter selections. Codes are available at https://github.com/emecercelik/Multi-frame-3D-detection.git.

INDEX TERMS 3D object detection, KITTI, LiDAR, multi-frame fusion.

I. INTRODUCTION

Perception is an important step for autonomous systems to plan their actions safely, especially in dynamic environments. For example, the dynamic nature of the traffic makes it vital for automated vehicles to obtain high-quality 3D detections to react to changes at appropriate times. Therefore, 3D object detection has gained increasing importance in the intelligent vehicle domain. Recently, large-scale benchmarks [1]–[3] with annotated lidar, RGB camera, and radar data have been introduced to compare different approaches.

The best performing state-of-the-art 3D detectors mostly rely only on lidar scans [4]–[6]. Several monocular camera-lidar fusion strategies [7]–[12] have been proposed to achieve robust 3D detection. However, fusion methods suffer from multi-modal feature [13] alignment problems.

The associate editor coordinating the review of this manuscript and approving it for publication was Mehul S. Raval.

State-of-the-art 3D object detectors rely on single-frame data [14]–[17]. However, a single-frame lidar point cloud can contain occluded or partially observed objects. These occlusions are especially prevalent in dense, urban cityscapes. In addition, the sparsity of lidar point clouds increases with distance [18]. This sparsity causes inconsistent single-frame detection performance. Against this backdrop, we propose a multi-frame approach to compensate for the lack of information by using object features already obtained in previous frames (Fig. 1).

Multi-frame approach has been studied quite intensively for 2D image-based problems. There have been attempts to fuse object-level features [19]–[24] and scene-level features [25]–[29] to improve the 2D detection quality. Also, action recognition studies rely on multi-frame information to reach more reliable decisions [30], [31], since one scene in a single-frame can be related to multiple actions without observing corresponding scenes in the prior frames. Multi-frame processing has been also studied in the context of

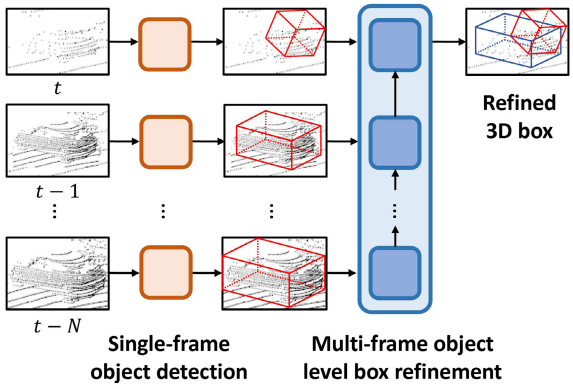


FIGURE 1. Single-frame 3D detectors rely only on the point cloud available in one frame. Therefore, quality of 3D detections decreases for far away or occluded objects, which reflect only a small number of points. We propose fusing object-level features in multiple frames to compensate for the missing information in the current frame. In this way, we utilize features of high-quality detections from the previous frames for poor-quality features in the most recent scene.

tracking. Instead of comparing two frames [32], [33], methods utilizing more than two frames reach better 2D tracking capability [34]–[36].

3D object tracking methods work also on successive frames to track objects [37], [38]. It has been shown that using object-level features improves the tracking quality further [39]. However, processing multi-frame data for 3D object detection has been limited until the release of large-scale datasets [1], [2]. Recently, there are studies considering scene-level temporal fusion [40]–[42], however these methods suffer from feature alignment problems in two successive frames due to the movement of objects.

In this study, we propose an object-level temporal fusion method to improve 3D object detection results. We extend Frustum PointNet architecture [43] with our temporal fusion module, which fuses features of the same object from successive frames to obtain a more representative feature. The Frustum PointNet generates object-level features to predict 3D bounding boxes using segmented frustum points. We fuse the object-level feature of an object in the current frame with its features from the previous frames temporally to predict a more accurate 3D bounding box. We run our method on the KITTI Multi-object Tracking Benchmark dataset since it contains sequential data contrary to KITTI 3D object detection dataset. The validations are done for car, pedestrian, and cyclist classes. We compare out method with the vanilla Frustum PointNet baseline as well as the state-of-the-art object detectors that apply RGB-Lidar fusion. Comparing to the Frustum PointNet baseline, we reach 7%, 4%, and 6% improvement on car, pedestrian, and cyclist classes in moderate difficulty, respectively.

We list our main contributions below:

- A novel multi-frame 3D object detection strategy to increase consistency of bounding box predictions by fusing object-level features from multiple frames.
- An *object-level* RNN-based temporal RGB-Lidar fusion approach for 3D object detection. We investigate the temporal fusion strategies for the RNN and provide our results through extensive ablation studies.
- Experimental validation on a commonly used benchmark. We provide our 3D detection results on the KITTI object tracking.

II. RELATED WORK

A. SINGLE-FRAME RGB-LIDAR FUSION FOR 3D OBJECT DETECTION

Multi-modal perception systems benefit from redundancy for alleviating sensor failure problems. RGB-lidar fusion is commonly used for multi-modal 3D object detection, which follows two main strategies: feature-level and high-level information fusion.

The common approach for feature-level method is the fusion of lidar bird's eye view (BEV) representations with RGB image features [10], [11]. Additionally, [7] and [13] combine RGB image features and 2D segmentation outcomes with lidar features using calibration information. Cross-modality fusion further improves the 3D detection quality [12]. As a high-level fusion method, 2D bounding boxes obtained from RGB images are used to extract frustums of objects to reduce the search space in the entire point cloud [8], [43]. Similarly, [9] uses 2D semantic segmentation results to filter out background points in the point cloud.

Our temporal fusion method is based-on the Frustum PointNet [43] architecture, which applies high-level RGB and lidar frustum fusion. In this way, we obtain object-specific features from the frustums of 2D bounding boxes. Single-frame methods are limited to the data of the current frame, which makes them prone to noisy data. However, our multi-frame approach can propagate features from previous frames to the current frame, which helps obtaining a wider view, richer features, and as a result more stable 3D detections.

B. 2D VIDEO OBJECT DETECTION AND TRACKING

Multi-frame processing has been studied more extensively for 2D object detection and tracking tasks than their counterparts in 3D. Even though, single-frame 2D object detection methods provide good results [44], [45], they can miss previously-detected objects in certain frames of a video. Aggregating object-level features through multiple frames alleviates this problem. References [19], [34] combines 2D regions in 3D tube-like volumes among multiple frames, [20] and [21] leverage attention mechanisms among multi-frame objects, [22] measures similarities and concatenates similar regions in the video, [23] and [24] refine object-level features using RNNs. On the other hand, several approaches to aggregate features of the entire scene have been proposed such as using convolutional LSTMs [25] and LSTMs [28], attention mechanisms [26], [29] and flow fields [27]. Similarly for the 2D tracking problem, [36] processes object-bounding tubes with 3D convolutions, [35] utilizes graph and motion features, and [46] and [47] use transformer architectures to track objects.

Even though object-level feature aggregation has been frequently and successfully considered for 2D video object detection, this was an untouched method for the 3D object detection problem. In this study, we propose multi-frame object-level feature propagation method is to obtain better 3D bounding boxes from successive lidar point clouds in time.

C. 3D OBJECT DETECTION AND TRACKING WITH LIDAR

1) SINGLE-FRAME 3D OBJECT DETECTION

Voxel-based and point-based feature learning methods are the two main methods to process unordered and sparse lidar point cloudes. Voxel-based methods learn features of the points in 3D grid cells, which are called voxels, by applying 3D convolutions [5], [48], [49] to voxels. References [50] and [51] utilize pillars, infinitely-high voxels, to increase the detection speed. Differently, point-based 3D detectors apply PointNets [52], [53] to learn features from sampled lidar points [16], [17]. To improve the representation quality, both methods have recently been combined either sequentially [54], in parallel [15], [55], [56], or one as an auxiliary network [6].

In the voxel-based approaches, an object can occupy multiple voxels, which makes it difficult to obtain and propagate object-level features. Similarly, point-based methods that rely only on lidar point clouds sample keypoints, also multiple of which can represent an object. Instead, we utilize and extend Frustum PointNet [43], which extracts features from the region of the object. Thus, object-specific features are obtained without additional computation and merging process.

2) 3D MULTI-OBJECT TRACKING

3D multi-object trackers also utilize 3D bounding box and appearance information from multiple successive frames. [38] tracks 3D bounding box of objects using a 3D Kalman filter. References [37] learns a similarity map on 2D appearance features from two frames. References [57] predicts also 2D and 3D center offsets to match objects in successive frames. [58] associates objects by learning geometry- and appearance-based costs for 3D bounding boxes of two adjacent frames. [59] utilizes LSTMs on monocular video frames. Additionally, [60] makes use of lidar appearance features. [39] employs 2D and 3D appearance and motion features of 3D bounding boxes detected in multiple time-steps.

Similar with the 3D tracking methods, we fuse 3D appearance features of objects from multiple frames to predict better 3D bounding boxes instead of associating objects. Thus, the more representative multi-frame object features can result in better 3D bounding box prediction comparing to using single-frame features.

3) MULTI-FRAME 3D OBJECT DETECTION

Multi-frame 3D object detection gained attention after the availability of datasets that provide sequences of data [1], [2]. Even though, KITTI multi-object tracking dataset [3] provides sequences of lidar scans, it has been only considered for the tracking task. The proposed methods for processing multiple frames have focused on scene-level aggregation so far. [61] utilizes motion maps from two successive frames. [62] uses 3D CNNs for multi-frame BEV features, whereas [63] and [40] apply convolutional LSTMs. Similarly, [41] proposes a custom-designed LSTM with spatial feature alignment between frames. Differently, non-local attention mechanisms [64] as well as transformer architectures [42], [65] have been utilized to build the spatio-temporal relation between multiple frames.

These methods show usefulness of multi-frame approaches to improve quality of 3D detections. However, scene-level feature aggregation requires spatial alignment of features between successive frames. Different from these approaches, we utilize object-level features from multiple frames to obtain stronger object representations without the need for an additional spatial alignment. Our previous work [66] includes also object-level temporal features on lidar-RGB fusion for 3D object detection, in which tracking and the extent of temporal associations were not considered. Here, we extend the previous study considering tracking of objects, ablation of multi-frame sequence lengths, comparison with other 3D detectors as well as a large number of ablations.

III. PROPOSED METHOD

Making use of the data from the previously-processed frames extends ego-vehicle's field of view and compensates for the occluded objects. With this idea, we aim to fuse an object's features in successive frames to obtain more informative object-specific features and finally its 3D bounding box as shown in Fig. 2. Therefore, we extend Frustum PointNet [43] v1 model, from which we can obtain object-specific features.

A. PROBLEM DEFINITION

Our method utilizes a point cloud $P^t = \{p_i | i = 1, \ldots, k\}$ and an RGB image $\mathbf{I}^t \in \mathbb{R}^{H \times W \times 3}$ sampled at time-step t, where k is the number of points, a single point $\mathbf{p}_i \in \mathbb{R}^3$, and H and W are height and width of the RGB image. A 2D detector detects a list of 2D bounding boxes $O^t = \left\{o_i^t | i = 1, \ldots, l\right\}$ using I^t, where $\mathbf{o}_i \in \mathbb{R}^4$ and l is the number of objects visible in I_t. A 2D tracker, h, takes O^t and O^{t-1} with the list of unique IDs of the objects, U^{t-1}, in frame $t-1$ to provide unique track IDs $U^t = \left\{u_i^t | i = 1, \ldots, l\right\}$ of O^t defined as $U^t = h(U^t, O^t, O^{t-1})$, where $\mathbf{u}_i \in \mathbb{Z}^+$.

Utilizing a τ length sequence of $S^t = \{(P^t, U^t, O^t), \ldots, (P^{t-\tau+1}, U^{t-\tau+1}, O^{t-\tau+1})\}$, we aim to obtain 3D bounding boxes $B^t = \left\{b_i^t | i = 1, \ldots, l\right\}$, where $\mathbf{b}_i^t \in \mathbb{R}^7$, with a 3D detector, which is capable of processing the multi-frame sequence S^t. We represent b_i^t with the width, height, and length, 3D center, and the orientation of the 3D bounding box. The whole pipeline is summarized in Fig. 2.

B. 2D DETECTION

Frustum PointNet extracts a subset of point cloud in the frustum of a 2D bounding box, o_i^t. Therefore, we require a 2D

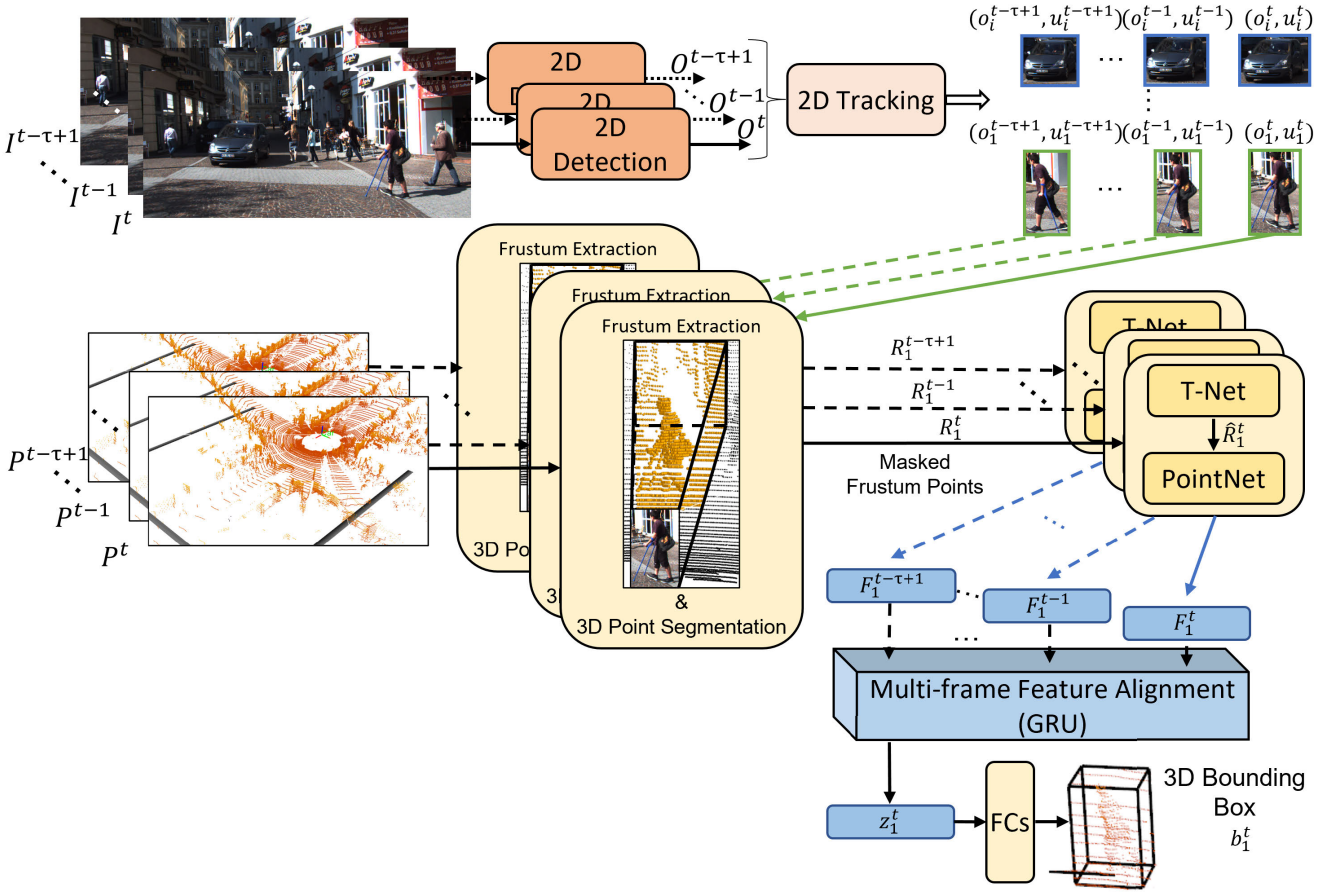


FIGURE 2. Multi-frame 3D detection pipeline: 2D bounding boxes and track IDs are obtained using a 2D detector and tracker, which are used for extracting frustum points (orangely-painted). For clarity, we show the process only for one tracked object shown with the 2D bounding boxes $\{o_1^{t-\tau+1}, \ldots, o_1^t\}$ and track IDs $\{u_1^{t-\tau+1}, \ldots, u_1^t\}$. From frustum points, 3D point segmentation module generates $\{R_1^{t-\tau+1}, \ldots, R_1^t\}$ that are predicted to be the object of interest's. Taking the masked points, T-Net and a PointNet generate object-level features. The object-level features $\{F_1^{t-\tau+1}, \ldots, F_1^t\}$ of the object $\{o_1^{t-\tau+1}, \ldots, o_1^t\}$ are fused in time with a GRU-based multi-frame feature alignment module. The final temporal feature (z_1^t) of the o_1^t is used to predict 3D bounding box parameters b_1^t.

detector g to obtain a set of 2D bounding boxes given as $O^t = g(I^t)$. Frustum PointNet utilizes a custom-designed detector for high recall, which is not included in the original Frustum PointNet repository. We provide our own 2D detectors in our repository. The 2D bounding boxes are used to decrease the search space in the entire point cloud. In addition, high recall is required not to miss the objects for 3D detection.

C. 2D TRACKING

We need track IDs of the 2D bounding boxes to associate features of the same object from multiple scenes. Therefore, we utilize a 2D tracker, which takes 2D bounding boxes in successive two frames, O^t and O^{t-1}, with unique track IDs U^{t-1} and outputs U^t as seen at the top of Fig. 2. Since the tracking problem was not considered in the original Frustum PointNet [43] study, we employed our 2D trackers to match objects in successive frames. As we explain our approach in subsection IV-D, we tested SORT [32] and Deep SORT [33] 2D trackers on top of 2D detections for assigning 2D bounding boxes to each other in successive frames.

D. SINGLE-FRAME FRUSTUM-LEVEL RGB-LIDAR FUSION

Frustum PointNet is a single-frame 3D detector, which takes RGB images and lidar point clouds as inputs and fuses them by extracting lidar points (P_i) in the frustum of each object's 2D bounding box o_i. The point set is decreased to $P_i = \{p_i | i = 1, \ldots, m\}$, where m is the number of points sampled randomly in the frustum. However, the quality of the object-level features and the 3D detection depends on the sampling strategy, which is done randomly in the original study [43]. We also follow the same procedure.

E. OBJECT-LEVEL SINGLE-FRAME 3D DETECTION

Frustum PointNet [43] consists of 3 main parts. The frustum proposal part extracts frustum lidar points, a subset of point clouds $P_{O^t} = \{P_1, P_2, \ldots, P_l\}$, using 2D bounding boxes, O^t. This leaves irrelevant lidar points outside of the frustum. However, there are still points in the frustum that do not belong to the object of interest, which would cause a noisy representation.

The 3D instance segmentation PointNet aims to minimize the irrelevant frustum points. It removes points with low objectness scores from the frustum point set (P_i^t). A masked subset (R_i^t) of P_i^t is obtained such that $R_i^t = \{p_i | i = 1, \ldots, n\}$ as shown in Fig. 2. n is the number of points after masking operation and satisfies $n < m$.

The amodal 3D box estimation module consists of a T-Net and an amodal 3D box estimation PointNet. The T-Net takes R_i^t and estimates center residuals, which are used to translate R_i^t to a point set $\hat{R}_i^t$ in the new coordinate system. The amodal 3D box estimation PointNet takes $\hat{R}_i^t$ and outputs global feature (F_i^t) of the object o_i^t using multi-layer perceptrons (MLPs) and a max-pool operation. F_i^t is further processed by fully-connected layers to predict 3D box parameters b_i^t.

F. TEMPORALLY FUSING OBJECT-LEVEL FEATURES FOR REFINING 3D DETECTION

The global feature F_i represents the position, orientation, and shape of the object o_i in an abstract way. However, the o_i can be only partially observed by the lidar sensor at a time-step, which causes dramatic changes in F_i. Therefore, detection quality depends on the sampled $\hat{R}_i$ and the occlusion state of the o_i.

To alleviate the stated problem, we propose to use an object's features from successive frames to compensate for the loss of information and to obtain a richer representation. Our multi-frame feature alignment module, f, fuses global features of the same object from multiple frames given with $F_i^{[t-\tau+1,t]} = \{F_i^{t-\tau+1}, \ldots, F_i^t\}$ to obtain a more representative feature of the object as shown with $z_i^t = f(F_i^{[t-\tau+1,t]})$, where τ is the number of frames, as depicted at the bottom of Fig. 2.

We realize f with the gated recurrent units (GRUs) to fuse object-level features in time. The resulting feature vector from GRU cells is further processed with a fully-connected layer. We expect that the temporal feature vector z_i^t provides a better representation of the object, which is shown with the experiments in V. The T-Net shown in the Fig. 2 predicts center residuals $\hat{R}_i^t$ for the bounding box of the o_i^t. Adding this information to the object feature vector F_i^t would help the network to understand the center shifts of the bounding box. Therefore, we concatenate the center residuals from T-Net with the F_i^t to improve the representation quality. This choice is also experimented as given in IV-D.

G. MULTI-FRAME FEATURE ALIGNMENT STRATEGIES

The obtained temporal feature z_i^t can be used with different multi-frame feature alignment strategies as shown in Fig. 3 for predicting 3D bounding boxes. The straightforward approach would be to feed z_i^t directly to the FCs to predict 3D box parameters. We name this strategy as one branch (OB) seen in Fig. 3-a. Even though z_i^t is expected to contain a richer representation, it is still beneficial to have object's current feature vector, F_i^t, that is related more to the current position. Therefore, we combine the temporal feature z_i^t with the object feature F_i^t at time-step t through a *mean* operation, which is named as two branch (TB) in Fig. 3-b. In this way, the feature has also the awareness for the current state's predictions. We realized the mean operation by adding two feature vectors and dividing by 2.

Due to objects' movement, only objects' shape remains same between frames. As a third alignment option, we use z_i^t to predict shape parameters of the object bounding box and F_i^t to predict the rest of the parameters. To construct the final output, the parameters are concatenated. In addition, using temporal feature z_i^t for the shape prediction acts as an auxiliary loss for the preceding shared layers of F_i^t and z_i^t. This also ensures to obtain a more representative object feature vector F_i^t from the MLPs. This is called as Ours (seen in Fig. 3-c) throughout the experiments and results sections. None of the three strategies requires extra learnable parameters and the shape of the FCs are the same. As the ablation results in subsection V-B3 indicate, the Ours version provides considerably better results than the other two strategies.

IV. EXPERIMENTS

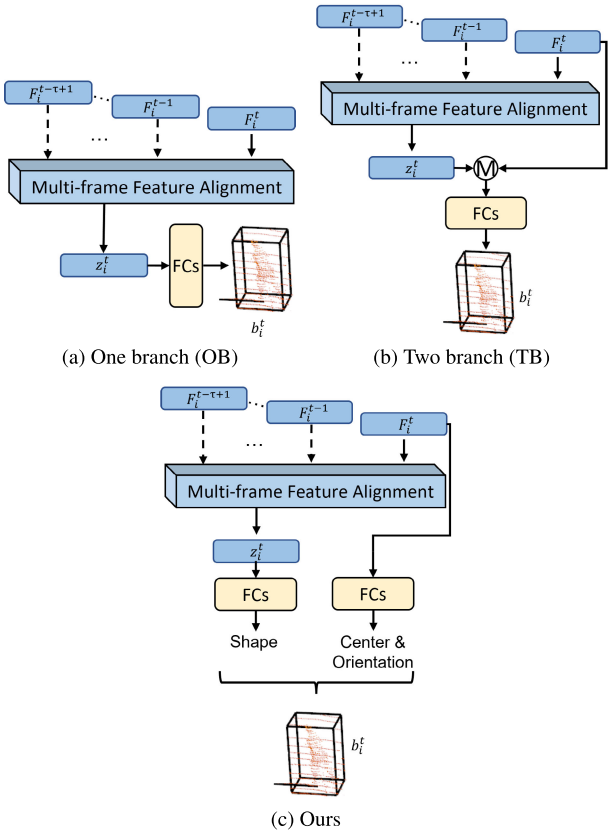
A. KITTI TRACKING DATASET

In this study, we use KITTI Multi-object Tracking Benchmark dataset [3], which provides sequential RGB images and lidar scans. We split the 21 drives of the dataset in training and validation sets. The validation split consists of drives 11, 15, 16, and 18 and the rest of the drives are used for the training. There are 6264 frames in the training and 1239 frames in the validation set. We tried to keep the ratio of object instances in the training and validation splits similar. In addition, 92% and 96.3% of the objects are visible in more than 10 frames in the training and validation sets, respectively. We choose the validation splits to be challenging. As it can be seen from Fig. 4, there are scenes from the urban and suburban areas. We also observe permanent occlusions that occur due to the parked cars, pedestrians crossing in front of the vehicles, the traffic jam, and overtaking vehicles. In addition, we check the number of points inside object 3D bounding boxes as given in Table 1. In all difficulty levels defined for KITTI, the mean number of points inside objects decreases with the distance considerably. Also, comparing mean number of points for the training and validation splits, validation split seems to be more challenging since the objects contain less points in average. The dashes for the easy difficulty level indicate that there is no object in that distance bin.

B. METRICS

We utilize average precision (AP) metric to measure the performance of the network as it is recommended in the KITTI 3D object detection benchmark.[1] The AP is calculated through 40 recall points as also stated in [67]. IoU threshold is 0.7 for car class and 0.5 for pedestrian and cyclist classes for AP calculation.

[1] http://www.cvlibs.net/datasets/kitti/eval_object.php



(a) One branch (OB) (b) Two branch (TB)

(c) Ours

FIGURE 3. Multi-frame feature alignment strategies for the temporal features.

TABLE 1. Mean number of lidar points in object 3D bounding boxes according to the distance and KITTI difficulty level.

Training split	0-35 m	35-50 m	50 m +
Easy	375.4	78	-
Moderate	209.1	36.8	17.6
Hard	95.6	25.3	8.3
Validation split	0-35 m	35-50 m	50 m +
Easy	304.4	-	-
Moderate	137.3	31.7	13.5
Hard	93.7	16.4	5.25

C. LOSS FUNCTION

We utilize the original multi-task loss function used in Frustum PointNet [43], which consists of mask, center, heading class, size class, heading regression, size regression, the T-Net center regression, and corner losses shown with L_{mask}, L_{center}, L_{h-c}, L_{s-c}, L_{h-r}, L_{s-r}, L_{T-c}, and L_{corner} respectively. The classification losses are realized with softmax loss and the regression ones are with the smooth-L1 loss. We add a cosine distance loss between the object-level features of an object in the current (F_i^t) and the preceding (F_i^{t-1}) frames to the multi-task loss of the Frustum PointNet. This loss function aims to force the network to form the feature in the previous frame with a different set of points. The cosine distance loss is shown in Eq. 1, where the υ and the ω are feature vectors, between which the distance is measured.

$$L_{cos}(\upsilon, \omega) = 1 - \frac{\sum_{k=1}^{l} \upsilon_k \omega_k}{\sqrt{\sum_{k=1}^{l} \upsilon_k^2} \sqrt{\sum_{k=1}^{l} \omega_k^2}} \quad (1)$$

The multi-task loss function is constructed as shown in Eq. 2, where α, β, and γ are the weights of the 3D box losses, corner loss, and the cosine distance loss.

$$L_{total} = L_{mask} + \alpha(L_{center} + L_{h-c} + L_{s-c} + L_{h-r} + L_{s-r} + L_{T-c} + \beta L_{corner}) + \gamma L_{cos} \quad (2)$$

D. EXPERIMENTS & ABLATIONS

We compare 3D AP results of multi-frame object-level temporal fusion with the Frustum PointNet baseline and state-of-the-art 3D detectors on KITTI tracking dataset for car, pedestrian, and cyclist classes. In addition, we conduct ablation studies to investigate validity of the results.

Our method requires 2D bounding boxes and unique track IDs of boxes. For the state-of-the-art comparison, we utilize perturbed ground-truth 2D detections and ground-truth track IDs for the baseline Frustum PointNet and also for our method. We conduct an ablation study to show method's performance with the predicted track IDs obtained by SORT [32] and Deep SORT [33] 2D tracking methods. We also study effects of different sequence lengths (τ). $tau = 1$ means using a fully-connected layer instead of GRUs to keep a similar depth. Third ablation discusses results of feature alignment strategies explained in III-G as well as results of training with the cosine distance loss on the proposed strategies. As a fourth ablation, we compare GRU-based fusion with LSTM-based fusion and a simple convolution-based temporal fusion. In the convolution-based fusion, we use two convolutional layers to obtain the temporal feature vector (z_i^t) instead of a recurrent layer. Our fifth ablation compares the results according to the depth of features to be aggregated temporally. The Frustum PointNet max-pools the point features in its Amodal 3D Box Estimation PointNet and concatenates the max-pooled features with the k-length class vector. This is called as the *global* feature. Two FC layers are added after the *global* feature. We call the output of the first FC layer as $fc1$ and use it in all our experiments as the temporal feature vector (z_i^t). In this ablation, we also take the *global* features as z_i^t and compare with the $fc1$ features. Finally, we study the extent of features by temporally fusing scene-level features. We compare the scene-level fusion with our object-level fusion results.

E. IMPLEMENTATION DETAILS

The object-specific global feature in the amodal 3D box estimation PointNet is processed with two FC layers and an output layer for the prediction of the box parameters. The FC layers have (512,256) units. We use output of the first FC layer (512) as F_i^t. The features of the same object from the previous time steps ($\{F_i^{t-\tau+1}, \ldots, F_i^{t-1}\}$) are kept in the memory and fed during training. The rest of the network is kept the same as the v1 version of the Frustum PointNet.

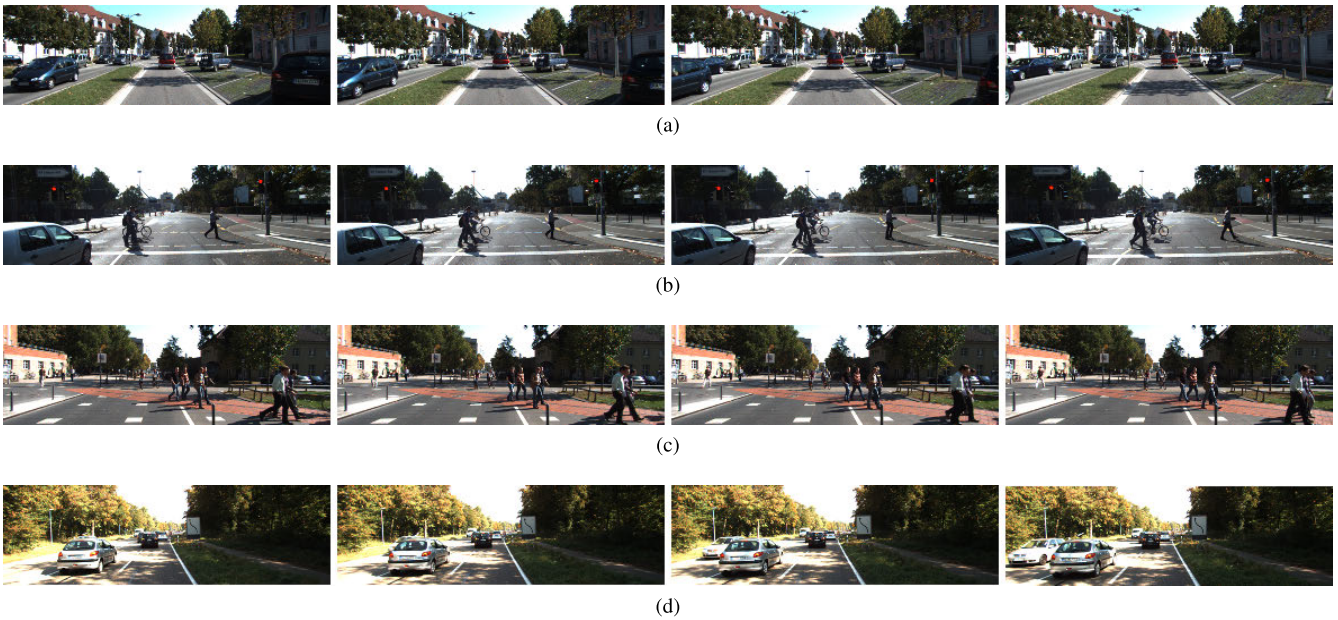


FIGURE 4. Successive scenes from our validation ssplit of the KITTI multi-object tracking dataset. (a) Drive 11, (b) Drive 15, (c) Drive 16, (d) Drive 18.

TABLE 2. KITTI 3D AP results pedestrian (IoU = 0.5) and cyclist (IoU = 0.5) classes on KITTI tracking validation drives.

Method	Pedestrian 3D AP			Cyclist 3D AP		
	Easy	Moderate	Hard	Easy	Moderate	Hard
SECOND [49]	57.1	56.3	49.4	83.1	57.2	54.6
PointPillars [50]	54.8	54.3	48.2	**86.7**	43.9	43.2
MVXNet [7]	38.5	40.9	35.2	68.6	32.4	32.6
AVOD-FPN [10]	32.8	33.6	32.3	21	13.1	9.1
Frustum PointNet v1 [43]	56.8	56.5	50.4	81.3	59.4	58.8
Ours	**67.5**	**60.4**	**53.6**	82.5	65.3	65
Ours (w/ Cent)	64.5	59.4	52.7	85.1	**74.5**	**67.3**

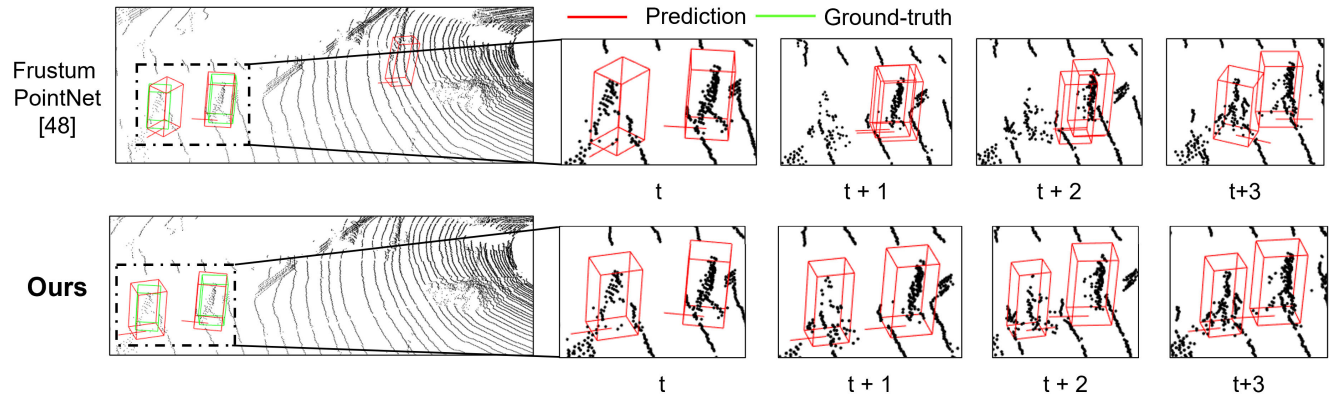


FIGURE 5. Qualitative comparison of the pedestrian class between the baseline Frustum PointNet [43] and the proposed method (Ours) for successive frames (Green: Ground-truth, Red: Detection). For simplicity, we only show the detection (red boxes) for the zoomed-in crops. The baseline can detect the pedestrians as shown in the dashed rectangular at time-step *t*. However, it confuses the localization of the bounding boxes in the following frames. On the other hand, our method can detect and keep the 3D boxes correctly in all the scenes.

We also utilize the same training hyperparameters as the Frustum PointNet. The number of points in a frustum is fixed and taken as 1024, which is randomly sampled. We train each parameter set for at least 5 times and take the best resulting checkpoint for evaluation. All the training and validation steps took place in a Docker container, which runs Python 3.6 and Tensorflow 1.15, and on a single Nvidia RTX 2080 GPU with an Intel Xeon E3-1225 v5 3.30GHz CPU.

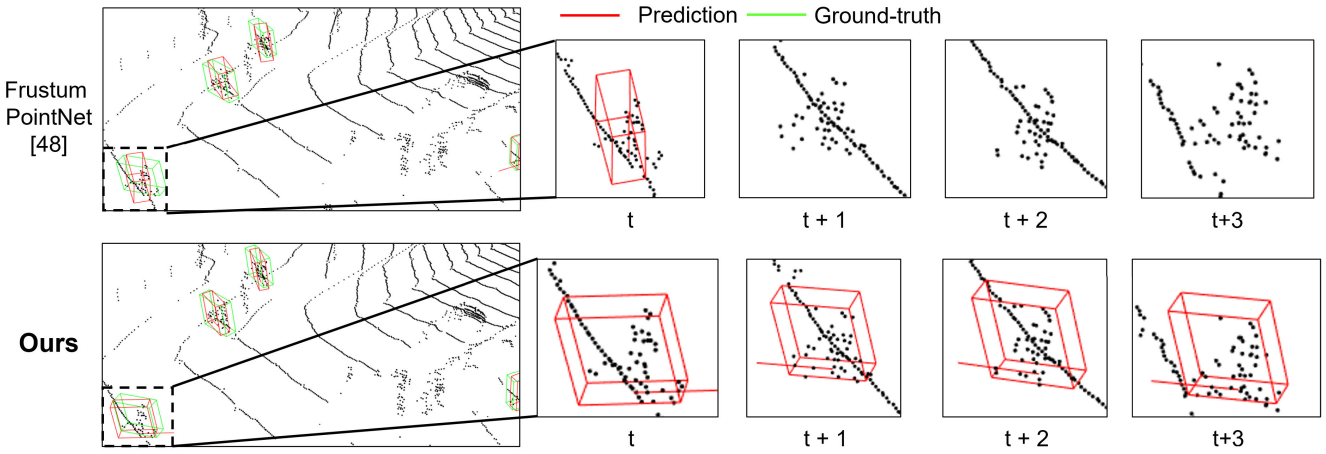


FIGURE 6. Qualitative comparison between the baseline Frustum PointNet [43] and Ours with temporal fusion on cyclist class (Green: Ground-truth, Red: Detection). We omit the ground-truth boxes in the zoomed-in crops for simplicity. Even though the Frustum PointNet was able to localize the cyclist correctly at time-step *t*, it misses the object in the upcoming frames. However, Ours with the temporal fusion can localize the object and keep the bounding box in the successive time-steps correctly.

TABLE 3. Tracking ablation results for three classes on KITTI tracking validation drives.

Gt 2D Det +	SORT			Deep SORT			Gt Tracking		
Car	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
Ours	**60.5**	**53.6**	**48.7**	58.9	52.1	47.3	83.1	72.2	65.1
Ours (w/ Cent)	60.2	52.9	48.2	58.1	51.1	46.7	83.7	65.6	64.1
Pedestrian	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
Ours	27.2	26.2	22.7	28.3	26.9	23.2	67.5	60.4	53.6
Ours (w/ Cent)	30.6	34.1	30.3	**31.6**	**34.9**	**30.9**	64.5	59.4	52.7
Cyclist	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
Ours	42	41.1	41.7	34.6	37.5	38.4	82.5	65.3	65
Ours (w/ Cent)	**55.4**	**61.5**	**55.5**	51.4	59.2	53.2	85.1	74.5	67.3

TABLE 4. KITTI 3D AP results car (IoU = 0.7) class on KITTI tracking validation drives.

Method	Car 3D AP		
	Easy	Moderate	Hard
MVXNet [7]	**88.4**	59.8	57.5
AVOD-FPN [10]	59.2	43.6	38.4
Frustum PointNet v1 [43]	83.1	65.4	**65.1**
Ours	83.1	**72.2**	**65.1**
Ours (w/ Cent)	83.7	65.6	64.1

V. RESULTS & DISCUSSION

In this section, we show and discuss the efficacy of the proposed object-level temporal feature fusion for 3D object detection task. We first share our quantitative and qualitative results and then provide the ablation results explained in IV-D.

A. COMPARISON OF 3D DETECTION PERFORMANCE

We compare our method with other 3D detection architectures and with the baseline Frustum PointNet [43] on the KITTI tracking validation set. As seen from Table 2, our method outperforms the compared detectors in the moderate difficulty for pedestrian and cyclist classes. In this table, we also provide our results for Ours (w/ Cent), which utilizes object-level features extended with the center prediction from T-Net as explained in III-F. Adding centers improves the 3D AP for the cyclist class. Cyclist is the least-represented class among all classes and we think that adding extra information helps network to learn how to localize the cyclist objects. For the car class, we also outperform the baseline in the moderate difficulty given in Table 4.

Our qualitative results indicate how our method outperforms the Frustum PointNet baseline for the far-away or occluded objects, which reflect a small number of points comparing to the closer objects. In Figure 5, Frustum PointNet misses the previously-detected pedestrians in the following frames. In this case, there are two pedestrians approaching each other, which causes the detector to miss the farther-away pedestrian. However, our method can keep the localization of the 3D box correctly. Figure 6 and 7 show that our method can detect the far-away cyclist and car objects quite accurately in all frames, respectively. However, Frustum PointNet without temporal fusion suffers from the localization problem and misses the previously-detected objects in the successive frames.

B. ABLATION STUDIES

1) COMPARISON FOR 2D TRACKING PERFORMANCE

The 3D detection performance of our method highly depends on the 2D tracking accuracy used to match object-level

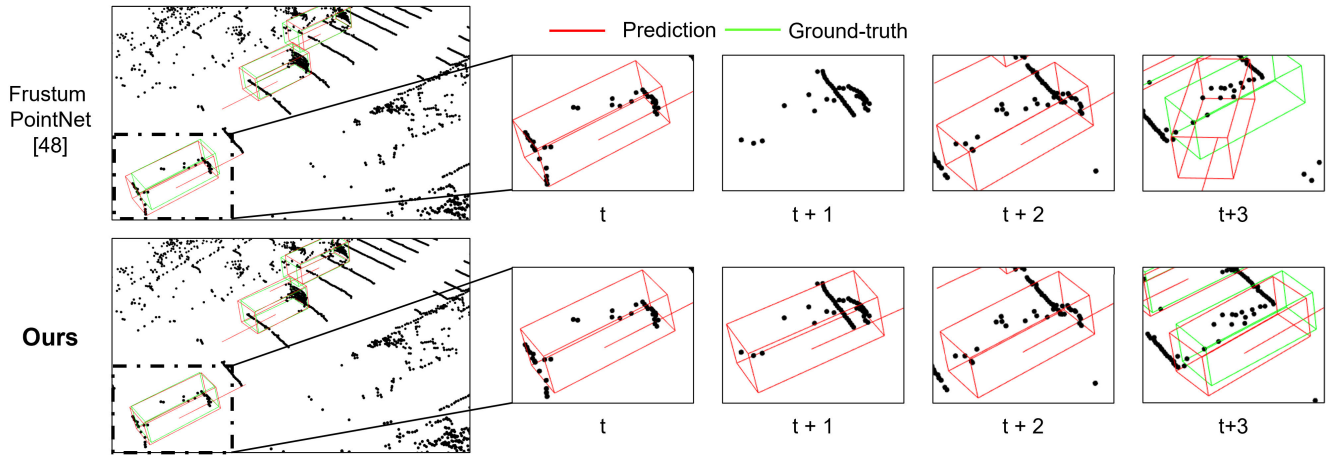


FIGURE 7. Qualitative comparison of 3D detection results between the baseline Frustum PointNet [43] and Ours with temporal fusion on car class (Green: Ground-truth, Red: Detection). The baseline detects the object in the first frame, but misses the object afterwards. However, our temporal fusion model can detect and keep the detected box in the successive frames as well. For simplicity, we show only the detected boxes (red) in the crops t, $t+1$, and $t+2$. At $t+3$, the baseline cannot predict the orientation correctly, whereas our method (Ours) can predict the heading consistently.

TABLE 5. Tau ablation results for three classes on KITTI tracking validation drives.

		Pedestrian 3D AP			Cyclist 3D AP			Car 3D AP		
	Tau	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
Frustum PointNet v1	-	56.8	56.5	50.4	81.3	59.4	58.8	83.1	65.4	65.1
Ours	1	52.6	47.6	46.7	80	56.6	55.5	77.1	62.1	55.5
Ours	2	54.9	50.6	49.4	77.3	64.5	64	83	65	63.7
Ours	3	**67.5**	**60.4**	**53.6**	82.5	65.3	65	83.1	**72.2**	65.1
Ours	4	58.1	52.4	50.9	82.8	58.1	57.5	83.5	64.7	64
Ours	5	52.5	49.8	48.6	82.8	63.8	57.8	**85.5**	65.2	63.6
Ours	6	56.4	50.7	49.4	**83.1**	57.3	56.6	80.9	64.8	63.7
Ours	7	53.8	48.9	42.7	78.8	62.3	56.1	82.8	66.3	**65.3**
Ours	8	56.9	50.7	49.2	81	**66.1**	**65.1**	79.2	64.3	63.1
Ours	9	51.5	47.7	41.6	78.8	52.9	47.5	80.7	64.4	63.2
Ours	10	55.2	49.4	43.6	76.4	56.6	56.2	80.7	64.4	63.2
Ours	15	51.7	47.2	41.2	83	56.5	55.8	82.4	65.8	64.1

features in time. Therefore, we also evaluate our method using SORT [32] and Deep SORT [33] 2D trackers. We train our method with the ground-truth track IDs and evaluate on the KITTI validation set with predicted track IDs. Results are given in Table 3. Comparing to the ground-truth tracking, the AP values decrease for all classes and difficulty levels, which would also require fine-tuning of the trained network with the predicted track IDs.

2) COMPARISON OF SEQUENCE LENGTH

We evaluate our results with different sequence lengths as seen in Table 5. The best results are obtained with $\tau = 3$ for pedestrian and car classes, however we obtained the best score for the cyclist class with $\tau = 8$. Comparing $tau > 1$ with $tau = 1$ results, the improvement originates from the temporal fusion, but not from the additional depth in the architecture comparing to the Frustum PointNet baseline.

3) COMPARISON OF THE FEATURE ALIGNMENT STRATEGIES

We also validate our choice of placement of the temporal fusion module in the architecture in Table 6. The best results are obtained mostly using Ours version. In this strategy, training the network with the temporal features acts as an auxiliary loss for the prediction of the center and the orientation since the feature in the current frame is shared. This helps learning more representative features for the shared layers.

In Table 7, we also provide training results using the cosine loss on the object-level features from two subsequent frames. Except the car class for Ours, training with cosine loss improves the results for all fusion strategies.

4) COMPARISON OF TEMPORAL FUSION TYPE

GRU is our main choice for multi-frame fusion. We also test using convolutional layers and LSTM layers instead of GRU separately as explained in IV-D. As shown in Table 8, GRU-based feature aggregation outperforms convolution-based method for all classes in the moderate difficulty level. However, LSTM provides better results for all difficulty levels of the Cyclist class even though GRU performs better than LSTM for Car and Pedestrian classes.

5) COMPARISON OF FEATURE DEPTH

We also evaluate the performance of our multi-frame alignment method according to the depth of features. As explained

TABLE 6. Ablation of the feature alignment strategies: AP on KITTI validation for three classes.

Car	τ	Easy	Moderate	Hard
Ours	3	83.1	**72.2**	**65.1**
Ours (w/ Cent)	4	**+0.6**	-6.6	-1
OB	3	-3.2	-10.5	-4.9
TB	4	-0.9	-9.7	-3.4
Pedestrian	τ	Easy	Moderate	Hard
Ours	3	**67.5**	**60.4**	**53.6**
Ours (w/ Cent)	3	-3	-1	-0.9
OB	6	-9.7	-9.1	-3.8
TB	4	-9.8	-8.6	-2.9
Cyclist	τ	Easy	Moderate	Hard
Ours	3	84.9	65.3	65
Ours (w/ Cent)	4	**+1.3**	**+9.2**	**+2.3**
OB	4	-7.4	-14.6	-14.2
TB	4	-4.4	-1.6	-1.9

TABLE 7. Ablation of training with cosine loss for feature alignment strategies: AP on KITTI validation for three classes.

		Cosine Loss		
Car	τ	Easy	Moderate	Hard
Ours	4	**82.5**	65.5	**64.1**
Ours (w/ Cent)	3	-1.3	**+0.1**	-0.2
OB	3	-3.2	-3.4	-3
TB	3	-1	-1.7	-1.5
Pedestrian	τ	Easy	Moderate	Hard
Ours	4	65.7	**63.3**	**56.4**
Ours (w/ Cent)	4	-0.1	-4.2	-3.8
OB	6	+0.3	-4.2	-3.8
TB	6	**+2.3**	-2.7	-2.4
Cyclist	τ	Easy	Moderate	Hard
Ours	3	85.6	65.9	65.3
Ours (w/ Cent)	6	**+1.9**	**+10.9**	**+4.4**
OB	3	-6.1	-1.5	-1.3
TB	3	-3.3	-0.6	-0.8

TABLE 8. Ablation of layer type for multi-frame fusion AP on KITTI object tracking validation set for three classes.

Temporal Fusion Type	Ours		
Car	Easy	Mod	Hard
GRU	83.1	**72.2**	**65.1**
GRU (w/ Cent)	83.7	65.6	64.1
Conv	82.5	64.6	63.3
Conv (w/ Cent)	**84.6**	65.2	63.7
LSTM	83.5	65.7	64.6
LSTM (w/ Cent)	83.1	65.7	64.4
Pedestrian	Easy	Mod	Hard
GRU	**67.5**	**60.4**	**53.6**
GRU (w/ Cent)	64.5	59.4	52.7
Conv	62.2	56.8	50.6
Conv (w/ Cent)	64.8	57.4	52.1
LSTM	64.3	52.7	51.4
LSTM (w/ Cent)	65.7	58.5	51.9
Cyclist	Easy	Mod	Hard
GRU	84.9	65.3	65
GRU (w/ Cent)	86.2	74.5	67.3
Conv	84.8	58.1	57.8
Conv (w/ Cent)	86.9	69	68.8
LSTM	**87**	69	68.4
LSTM (w/ Cent)	85.4	**76.5**	**69.4**

in IV-D, we use *global* features as the object-specific feature vectors instead of using the output of the first FC layer, $fc1$. *global* features have a larger dimension than $fc1$ features and

TABLE 9. Ablation of feature depth AP on KITTI object tracking validation set for three classes.

Feature Length	*global*			$fc1$		
Car	Easy	Mod	Hard	Easy	Mod	Hard
Ours	82.9	65.4	64.1	83.1	**72.2**	**65.1**
Ours (w/ Cent)	82.4	65.2	63.6	**83.7**	65.6	64.1
Pedestrian	Easy	Mod	Hard	Easy	Mod	Hard
Ours	68.8	62.5	55.6	67.5	60.4	53.6
Ours (w/ Cent)	**69.9**	**62.8**	**55.8**	64.5	59.4	52.7
Cyclist	Easy	Mod	Hard	Easy	Mod	Hard
Ours	**89**	76.6	76.4	84.9	65.3	65
Ours (w/ Cent)	88.4	**77.6**	**77.3**	86.2	74.5	67.3

TABLE 10. Ablation of feature extent AP on KITTI object tracking validation set for three classes.

	Scene-level			Object-level		
Car	Easy	Mod	Hard	Easy	Mod	Hard
Ours	59.7	39	34.7	83.1	**72.2**	**65.1**
Ours (w/ Cent)	59.4	38.1	33.8	**83.7**	65.6	64.1
Pedestrian	Easy	Mod	Hard	Easy	Mod	Hard
Ours	34.2	31.6	27.6	**67.5**	**60.4**	**53.6**
Ours (w/ Cent)	34.6	32.2	28.1	64.5	59.4	52.7
Cyclist	Easy	Mod	Hard	Easy	Mod	Hard
Ours	39.4	46.8	46.1	84.9	65.3	65
Ours (w/ Cent)	40.8	48.8	43.4	**86.2**	**74.5**	**67.3**

they represent lower-level features of the objects comparing to the $fc1$ features. As the results in Table 9 indicate, $fc1$ performs better for the Car class. However, *global* features provide better results for Pedestrian and Cyclist classes. Pedestrian and Cyclist classes are mostly represented with a smaller number of points comparing to the Cars. We think that lower-level features might be required for the aggregation of features in multiple frames. Hence, the two classes are detected better using the *global* features for the multi-frame alignment.

6) COMPARISON OF FEATURE EXTENT

Our multi-frame alignment method makes use of object-level features to improve 3D object detection quality. We also compare our method with the scene-level multi-frame feature alignment. We obtain the temporal features using all of the points in the frustum instead of using object-level segmentation. Thus, the features represent a larger field. The results are given in Table 10. Comparing to our object-level fusion, detection accuracy decreases drastically for all classes. This result also indicates the importance of spatial alignment of features in a scene-level feature fusion.

VI. CONCLUSION

This study introduced a multi-frame RGB-lidar fusion framework to decrease 3D object detection inconsistency across multiple frames. The proposed method achieves this by

aggregating object-level features of the same object from multiple frames to improve 3D object detection quality.

Experimental validation shows that our approach increases the performance of already existing networks. Extending Frustum PointNet with the proposed temporal feature aggregation strategy improved the 3D detection performance by 6.5%, 4%, and 6% for car, pedestrian, and cyclist classes.

Future work can extend the proposed multi-frame detection framework with additional modalities such as radar. The robustness of other multi-modal fusion systems can be increased with the proposed temporal aggregation idea.

REFERENCES

[1] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "NuScenes: A multimodal dataset for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2020, pp. 11621–11631.

[2] P. Sun *et al.*, "Scalability in perception for autonomous driving: Waymo open dataset," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2020, pp. 2446–2454.

[3] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2012, pp. 3354–3361.

[4] C. He, H. Zeng, J. Huang, X.-S. Hua, and L. Zhang, "Structure aware single-stage 3D object detection from point cloud," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2020, pp. 14494–14503.

[5] J. Deng, S. Shi, P. Li, W. Zhou, Y. Zhang, and H. Li, "Voxel R-CNN: Towards high performance voxel-based 3D object detection," 2020, *arXiv:2012.15712*. [Online]. Available: http://arxiv.org/abs/2012.15712

[6] C. He, H. Zeng, J. Huang, X.-S. Hua, and L. Zhang, "Structure aware single-stage 3D object detection from point cloud," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2020, pp. 11873–11882.

[7] V. A. Sindagi, Y. Zhou, and O. Tuzel, "MVX-Net: Multimodal Voxelnet for 3D object detection," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 7276–7282.

[8] Z. Wang and K. Jia, "Frustum ConvNet: Sliding frustums to aggregate local point-wise features for amodal 3D object detection," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 1742–1749.

[9] 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

[10] 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.

[11] 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.

[12] M. Zhu, C. Ma, P. Ji, and X. Yang, "Cross-modality 3D object detection," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis.*, Jan. 2021, pp. 3772–3781.

[13] S. Vora, A. H. Lang, B. Helou, and O. Beijbom, "PointPainting: Sequential fusion for 3D object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2020, pp. 4604–4612.

[14] S. Pang, D. Morris, and H. Radha, "CLOCs: Camera-LiDAR object candidates fusion for 3D object detection," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 10386–10393.

[15] 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," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2020, pp. 10529–10538.

[16] Z. Yang, Y. Sun, S. Liu, and J. Jia, "3DSSD: Point-based 3D single stage object detector," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2020, pp. 11040–11048.

[17] W. Shi and R. Rajkumar, "Point-GNN: Graph neural network for 3D object detection in a point cloud," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2020, pp. 1711–1719.

[18] H. Zhang, D. Yang, E. Yurtsever, K. A. Redmill, and Ü. Özgüner, "Faraway-frustum: Dealing with LiDAR sparsity for 3D object detection using fusion," 2020, *arXiv:2011.01404*. [Online]. Available: http://arxiv.org/abs/2011.01404

[19] K. Kang, H. Li, T. Xiao, W. Ouyang, J. Yan, X. Liu, and X. Wang, "Object detection in videos with tubelet proposal networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 727–735.

[20] H. Deng, Y. Hua, T. Song, Z. Zhang, Z. Xue, R. Ma, N. Robertson, and H. Guan, "Object guided external memory network for video object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Oct. 2019, pp. 6678–6687.

[21] M. Shvets, W. Liu, and A. Berg, "Leveraging long-range temporal relationships between proposals for video object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Oct. 2019, pp. 9756–9764.

[22] H. Wu, Y. Chen, N. Wang, and Z.-X. Zhang, "Sequence level semantics aggregation for video object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Oct. 2019, pp. 9217–9225.

[23] S. B. S. Tripathi, Z. Lipton, and T. Nguyen, "Context matters: Refining object detection in video with recurrent neural networks," in *Proc. Brit. Mach. Vis. Conf. (BMVC)*, E. R. H. Richard, C. Wilson, and W. A. P. Smith, Eds. York, U.K.: BMVA Press, Sep. 2016, pp. 44.1–44.12, doi: 10.5244/C.30.44.

[24] Y. Lu, C. Lu, and C.-K. Tang, "Online video object detection using association LSTM," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 2344–2352.

[25] M. Zhu and M. Liu, "Mobile video object detection with temporally-aware feature maps," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 5686–5695.

[26] G. Bertasius, L. Torresani, and J. Shi, "Object detection in video with spatiotemporal sampling networks," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 331–346.

[27] X. Zhu, J. Dai, L. Yuan, and Y. Wei, "Towards high performance video object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7210–7218.

[28] G. Ning, Z. Zhang, C. Huang, X. Ren, H. Wang, C. Cai, and Z. He, "Spatially supervised recurrent convolutional neural networks for visual object tracking," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2017, pp. 1–4.

[29] J. Kim, J. Koh, B. Lee, S. Yang, and J. W. Choi, "Video object detection using object's motion context and spatio-temporal feature aggregation," in *Proc. 25th Int. Conf. Pattern Recognit. (ICPR)*, Jan. 2021, pp. 1604–1610.

[30] O. Köpüklü, X. Wei, and G. Rigoll, "You only watch once: A unified CNN architecture for real-time spatiotemporal action localization," 2019, *arXiv:1911.06644*. [Online]. Available: http://arxiv.org/abs/1911.06644

[31] C. Yang, Y. Xu, J. Shi, B. Dai, and B. Zhou, "Temporal pyramid network for action recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2020, pp. 591–600.

[32] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2016, pp. 3464–3468.

[33] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2017, pp. 3645–3649.

[34] K. Kang, H. Li, J. Yan, X. Zeng, B. Yang, T. Xiao, C. Zhang, Z. Wang, R. Wang, X. Wang, and W. Ouyang, "T-CNN: Tubelets with convolutional neural networks for object detection from videos," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 10, pp. 2896–2907, Oct. 2018.

[35] M. Kraus, S. M. Azimi, E. Ercelik, R. Bahmanyar, P. Reinartz, and A. Knoll, "AerialMPTNet: Multi-pedestrian tracking in aerial imagery using temporal and graphical features," in *Proc. 25th Int. Conf. Pattern Recognit. (ICPR)*, Jan. 2021, pp. 2454–2461.

[36] B. Pang, Y. Li, Y. Zhang, M. Li, and C. Lu, "TubeTK: Adopting tubes to track multi-object in a one-step training model," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2020, pp. 6308–6318.

[37] E. Baser, V. Balasubramanian, P. Bhattacharyya, and K. Czarnecki, "FANTrack: 3D multi-object tracking with feature association network," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2019, pp. 1426–1433.

[38] X. Weng, J. Wang, D. Held, and K. Kitani, "AB3DMOT: A baseline for 3D multi-object tracking and new evaluation metrics," 2020, *arXiv:2008.08063*. [Online]. Available: http://arxiv.org/abs/2008.08063

[39] X. Weng, Y. Wang, Y. Man, and K. M. Kitani, "GNN3DMOT: Graph neural network for 3D multi-object tracking with 2D-3D multi-feature learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2020, pp. 6499–6508.

[40] S. McCrae and A. Zakhor, "3D object detection for autonomous driving using temporal LiDAR data," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2020, pp. 2661–2665.

[41] R. Huang, W. Zhang, A. Kundu, C. Pantofaru, D. A. Ross, T. Funkhouser, and A. Fathi, "An LSTM approach to temporal 3D object detection in LiDAR point clouds," in *Computer Vision—ECCV 2020*. Glasgow, U.K.: Springer, Aug. 2020, pp. 266–282.

[42] J. Yin, J. Shen, C. Guan, D. Zhou, and R. Yang, "LiDAR-based online 3D video object detection with graph-based message passing and spatiotemporal transformer attention," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2020, pp. 11495–11504.

[43] 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.

[44] 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.

[45] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2961–2969.

[46] F. Zeng, B. Dong, T. Wang, X. Zhang, and Y. Wei, "MOTR: End-to-end multiple-object tracking with transformer," 2021, *arXiv:2105.03247*. [Online]. Available: http://arxiv.org/abs/2105.03247

[47] P. Chu, J. Wang, Q. You, H. Ling, and Z. Liu, "TransMOT: Spatial-temporal graph transformer for multiple object tracking," 2021, *arXiv:2104.00194*. [Online]. Available: http://arxiv.org/abs/2104.00194

[48] 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.

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

[50] 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.*, Jun. 2019, pp. 12697–12705.

[51] Y. Wang, A. Fathi, A. Kundu, D. A. Ross, C. Pantofaru, T. Funkhouser, and J. Solomon, "Pillar-based object detection for autonomous driving," in *Computer Vision—ECCV 2020*. Glasgow, U.K.: Springer, Aug. 2020, pp. 18–34.

[52] 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.*, Jul. 2017, pp. 652–660.

[53] 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. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 5105–5114.

[54] 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.*, Oct. 2019, pp. 1951–1960.

[55] P. Bhattacharyya and K. Czarnecki, "Deformable PV-RCNN: Improving 3D object detection with learned deformations," 2020, *arXiv:2008.08766*. [Online]. Available: http://arxiv.org/abs/2008.08766

[56] S. Shi, L. Jiang, J. Deng, Z. Wang, C. Guo, J. Shi, X. Wang, and H. Li, "PV-RCNN++: Point-voxel feature set abstraction with local vector representation for 3D object detection," 2021, *arXiv:2102.00463*. [Online]. Available: http://arxiv.org/abs/2102.00463

[57] X. Zhou, V. Koltun, and P. Krähenbühl, "Tracking objects as points," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2020, pp. 474–490.

[58] S. Sharma, J. A. Ansari, J. K. Murthy, and K. M. Krishna, "Beyond pixels: Leveraging geometry and shape cues for online multi-object tracking," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 3508–3515.

[59] H.-N. Hu, Q.-Z. Cai, D. Wang, J. Lin, M. Sun, P. Kraehenbuehl, T. Darrell, and F. Yu, "Joint monocular 3D vehicle detection and tracking," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Oct. 2019, pp. 5390–5399.

[60] W. Zhang, H. Zhou, S. Sun, Z. Wang, J. Shi, and C. C. Loy, "Robust multi-modality multi-object tracking," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Oct. 2019, pp. 2365–2374.

[61] T. Yin, X. Zhou, and P. Krahenbuhl, "Center-based 3D object detection and tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2021, pp. 11784–11793.

[62] 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.

[63] A. El Sallab, I. Sobh, M. Zidan, M. Zahran, and S. Abdelkarim, "Yolo4d: A spatio-temporal approach for real-time multi-object detection and classification from LiDAR point clouds," in *Proc. NIPS Workshop MLITS*, 2018.

[64] K. Kumar and S. Al-Stouhi, "Real-time spatial-temporal context approach for 3D object detection using LiDAR," in *Proc. VEHITS*, 2020, pp. 432–439.

[65] Z. Yuan, X. Song, L. Bai, Z. Wang, and W. Ouyang, "Temporal-channel transformer for 3D LiDAR-based video object detection for autonomous driving," *IEEE Trans. Circuits Syst. Video Technol.*, early access, May 21, 2021, doi: 10.1109/TCSVT.2021.3082763.

[66] E. Erçelik, E. Yurtsever, and A. Knoll, "Temp-Frustum Net: 3D object detection with temporal fusion," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, 2021.

[67] A. Simonelli, S. R. Bulo, L. Porzi, M. López-Antequera, and P. Kontschieder, "Disentangling monocular 3D object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Oct. 2019, pp. 1991–1999.

EMEÇ ERÇELIK (Member, IEEE) received the B.Sc. and M.Sc. degrees in electronics engineering from Istanbul Technical University, in 2013 and 2016, respectively. He is currently pursuing the Ph.D. degree with the Chair of Robotics, Artificial Intelligence and Real-Time Systems, Technical University of Munich (TUM).

He has been involved in different projects about neurorobotics and intelligent vehicles, in which spiking and artificial neural networks are used. His current research interests include 3D object detection, multi-frame data processing, and sensory fusion for intelligent vehicles.

EKIM YURTSEVER (Member, IEEE) received the B.S. and M.S. degrees from Istanbul Technical University, in 2012 and 2014, respectively, and the Ph.D. degree in information science from Nagoya University, Japan, in 2019.

Since 2019, he has been with the Department of Electrical and Computer Engineering, The Ohio State University, as a Research Associate. Currently, he is working on bridging the gap between the advances in the machine learning field and the intelligent vehicle domain. His research interests include artificial intelligence, machine learning, computer vision, reinforcement learning, intelligent transportation systems, and automated driving systems.

ALOIS KNOLL (Senior Member, IEEE) received the M.Sc. degree in electrical/communications engineering from the University of Stuttgart, Stuttgart, Germany, in 1985, and the Ph.D. degree *(summa cum laude)* in computer science from the Technical University of Berlin (TU Berlin), Berlin, Germany, in 1988.

He was with the Faculty of the Computer Science Department, TU Berlin, until 1993. He joined Bielefeld University, Bielefeld, Germany, as a Full Professor, where he has served as the Director for the Technical Informatics Research Group, until 2001. Since 2001, he has been a Professor with the Department of Informatics, Technical University of Munich (TUM), Munich. He was also on the Board of Directors of the Central Institute of Medical Technology, TUM (IMETUM). From 2004 to 2006, he was an Executive Director of the Institute of Computer Science, TUM. His research interests include cognitive, medical robotics, multiagent systems, data fusion, adaptive systems, multimedia information retrieval, model-driven development of embedded systems with applications to automotive software and electric transportation, and simulation systems for robotics and traffic. He was a member of the EU's Highest Advisory Board on Information Technology, Information Society Technology Advisory Group (ISTAG), and its subgroup on Future and Emerging Technologies (FET), from 2007 to 2009.