MSF: Motion-guided Sequential Fusion for Efficient 3D Object Detection from Point Cloud Sequences

Chenhang He* Ruihuang Li* Yabin Zhang Shuai Li Lei Zhang†
The Hong Kong Polytechnic University
{csche, csrhli, csybzhang, cssli, cslzhang}@comp.polyu.edu.hk

Abstract

Point cloud sequences are commonly used to accurately detect 3D objects in applications such as autonomous driving. Current top-performing multi-frame detectors mostly follow a Detect-and-Fuse framework, which extracts features from each frame of the sequence and fuses them to detect the objects in the current frame. However, this inevitably leads to redundant computation since adjacent frames are highly correlated. In this paper, we propose an efficient Motion-guided Sequential Fusion (MSF) method, which exploits the continuity of object motion to mine useful sequential contexts for object detection in the current frame. We first generate 3D proposals on the current frame and propagate them to preceding frames based on the estimated velocities. The points-of-interest are then pooled from the sequence and encoded as proposal features. A novel Bidirectional Feature Aggregation (BiFA) module is further proposed to facilitate the interactions of proposal features across frames. Besides, we optimize the point cloud pooling by a voxel-based sampling technique so that millions of points can be processed in several milliseconds. The proposed MSF method achieves not only better efficiency than other multi-frame detectors but also leading accuracy, with 83.12% and 78.30% mAP on the LEVEL1 and LEVEL2 test sets of Waymo Open Dataset, respectively. Codes can be found at https://github.com/skyhehe123/MSF.

1. Introduction

3D object detection [1, 2, 6, 7, 9, 14, 21, 27–29, 36] is one of the key technologies in autonomous driving, which helps the vehicle to better understand the surrounding environment and make critical decisions in the downstream tasks. As an indispensable sensing device in autonomous driving systems, LiDAR collects 3D measurements of the scene in the form of point clouds. However, LiDAR can only produce partial view of the scene at a time, and the sparse and incomplete representation of point clouds brings considerable challenges to the 3D object detection task. In practice, the LiDAR sensor will continuously sense the environment and produce a sequence of point cloud frames over time. The multi-frame data can provide a denser representation of the scene as the vehicle moves. Therefore, how to fuse these multi-frame point cloud data for more accurate object detection is worth deep investigation.

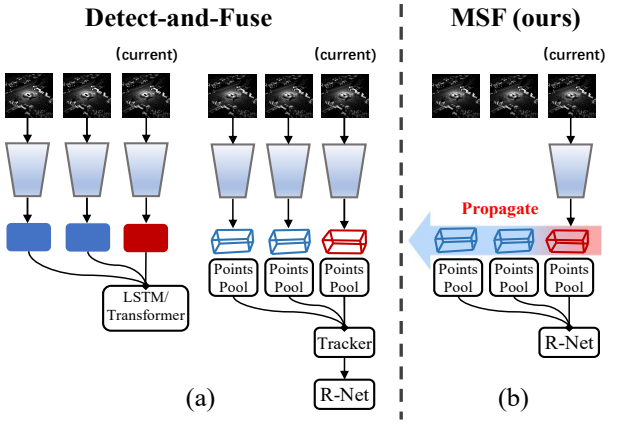

Figure 1. (a) The "Detect-and-Fuse" framework extracts features from each frame of the sequence and then fuses them, while (b) our proposed "Motion-guided Sequential Fusion" (MSF) method generates proposals on the current frame and propagates them to preceding frames to explore useful contexts in the sequence.

Recent works mainly focus on deep feature fusion with multi-frame point clouds, for example, aggregating dense birds-eye-view features via Transformer models [31, 37], passing the voxel features to LSTM [8] or GRU [32] modules for temporal modeling. Some top-performing detectors [2, 18] focus on fusing proposal features, where a tracker is employed to associate the 3D proposals across frames, and a region-based network is applied to refine the current proposals by incorporating contextual features from the proposal trajectories. These approaches generally follow a "Detect-and-Fuse" framework, as shown in Fig. 1(a), where the model requires to process each frame of the sequence,

*Equal contribution.
†Corresponding author.

and the predictions on the current frame rely on the results of preceding frames. Since online detection is a causal system, such a detection framework might cause significant delay if the network is still processing a preceding frame when the current frame is loaded.

In this paper, we propose an efficient Motion-guided Sequential Fusion (MSF) method, as shown in Fig. 1(b), which leverages the continuity of object motion to extract useful contexts from point cloud sequences and improve the detection of current frame. Specifically, considering that the motions of objects are relatively smooth in a short sequence, we propagate the proposals generated on current frame to preceding frames based on the velocities of objects, and sample reliable points-of-interest from the sequence. In this way, we bypass extracting features on each frame of the sequence, which reduces the redundant computation and reliance on the results of preceding frames. The sampled points are then transformed to proposal features via two encoding schemes and passed to a region-based network for further refinement. Specifically, a self-attention module is employed to enhance the interaction of point features within proposals, while a novel Bidirectional Feature Aggregation (BiFA) module is proposed to enforce the information exchange between proposals across frames. The refined proposal features consequently capture both spatial details and long-term dependencies over the sequence, leading to more accurate bounding-box prediction.

It is found that the existing point cloud pooling methods [2, 19, 23, 30] are inefficient, taking more than 40 milliseconds when processing millions of points from sequential point clouds. We find that the major bottleneck lies in the heavy computation of pair-wise distances between n points and m proposals, which costs $\mathcal{O}(nm)$ complexity. To further improve the efficiency, we optimize the point cloud pooling with a voxel sampling technique. The improved pooling operation is of linear complexity and can process millions of points in several milliseconds, more than eight times faster than the original method.

Overall, our contributions can be summarized as follows.

- An efficient Motion-guided Sequential Fusion (MSF) method is proposed to fuse multi-frame point clouds at region level by propagating the proposals of current frame to preceding frames based on the object motions.
- A novel Bidirectional Feature Aggregation (BiFA) module is introduced to facilitate the interactions of proposal features across frames.
- The point cloud pooling method is optimized with a voxel-based sampling technique, significantly reducing the runtime on large-scale point cloud sequence.

The proposed MSF method is validated on the challenging Waymo Open Dataset, and it achieves leading accuracy on the LEVEL1 and LEVEL2 test sets with fast speed.

2. Related Work

Single-frame 3D object detection. Recent research on single-frame 3D object detection is mainly focused on representation learning on point clouds. Voxel-based detectors [28, 33, 36] rasterize the point cloud into volumetric representation, followed by 3D CNN to extract dense features. Some works convert point clouds into 2D birds-eye-view [9] or range view [4, 11] representations, and process them with more efficient 2D CNN. Following PointNet++ [17], point-based methods [16, 23, 29, 30, 34] directly process point clouds in continuous space, and extract highly-semantic features through a series of downsampling and set abstraction layers. Voxel-point approaches [7, 12, 21] employ a hybrid representation, where the flexible conversion between voxel-based and point-based representations are explored, leading to better balance between efficiency and performance. Our method employs a high-quality voxel-based detector CenterPoint [33] as the proposal generation network to predict 3D proposals of current frame and their motions. We then employ an efficient region-based network to further refine these proposals by mining sequential points from point cloud sequence.

3D object detection from point cloud sequence. Multi-frame point clouds provide richer 3D information of the environment. While some single-frame detectors [3, 33] can be adapted to point cloud sequence by simply concatenating multi-frame point cloud as the input, the improvements are typically marginal and the performance can be even worse when encountering moving objects. Fast-and-furious [13] explores an intermediate fusion to align multi-frame point cloud by concatenating the hidden feature maps of the backbone network. However, it still suffers from the misalignment brought by the fast-moving objects in long sequence. Recent approaches [8, 32] demonstrate that an in-depth fusion can be achieved with recurrent networks. Unfortunately, the use of a single memory to store and update features across frames builds a potential bottleneck. To resolve such limitations, 3D-MAN [31] first attempts to employ the attention mechanism to align different views of 3D objects and then exploits a memory bank to store and aggregate multi-frame features for long sequence. Recently, Offboard3D [18] and MPPNet [2] improve much the detection performance, where they associate the detected boxes from each frame of the sequence as proposal trajectories, and extract high-quality proposal features by sampling sequential point cloud on the trajectories.

Our MSF method also samples points from the sequence, but it differs from those methods with proposal trajectories [2, 18] in that we only generate proposals on the current frame and propagate them to explore features in preceding frames. This makes our method much more efficient and favorable to online detection systems.

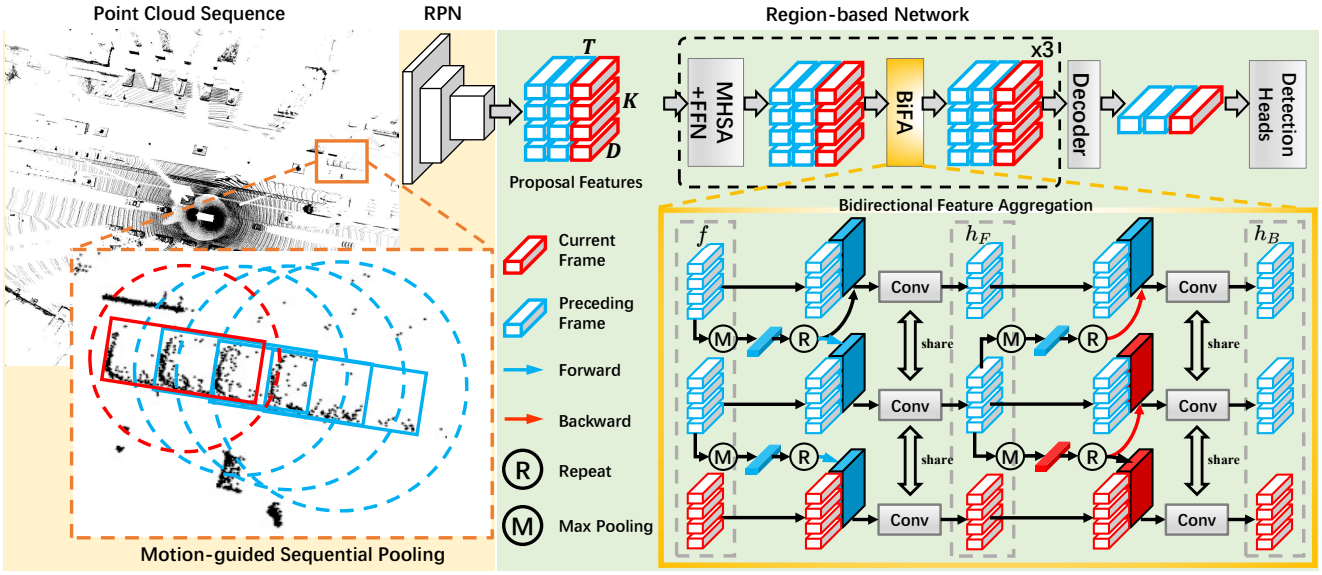

Figure 2. The overall architecture of our proposed Motion-guided Sequential Fusion (MSF) approach. By taking a point cloud sequence as input, MSF employs a region proposal network to generate proposals on the current frame and sample points-of-interest from the sequence by using motion-guided sequential pooling. The sampled points are encoded as high-dimensional proposal features and passed to a region-based network, where three learning blocks are consequently applied to refine the proposal features. A Bidirectional Feature Aggregation (BiFA) module is introduced in the region-based network to facilitate the interactions of proposal features across frames. The red and blue cubes represent single-point features from the current frame and preceding frame, respectively.

Table 1. Recall rates of foreground points by using per-frame detection based proposal trajectory method [2] and our motion guided proposal generation method. We employ the CenterPoint [33] as proposal generator and evaluate on Waymo validation split.

	4-frame	*8-frame*	*16-frame*
Trajectory [2]	93.2%	92.8%	90.5%
Ours ($\gamma = 1.0$)	92.3%	87.5%	78.3%
Ours ($\gamma = 1.1$)	93.5%	91.7%	87.3%

3. Motion-guided Sequential Fusion

This section presents our Motion-guided Sequential Fusion (MSF) approach for efficient 3D object detection on point cloud sequences. The overall architecture of MSF is illustrated in Fig. 2. In Sec. 3.1, we describe the details of motion-guided sequential pooling, which effectively mines reliable sequential points-of-interest based on the proposals of current frame. In Sec. 3.2, we present the region-based network, including the formulation of proposal features and a novel bidirectional feature aggregation module. In Sec. 3.3, we demonstrate a voxel-based sampling technique to accelerate the current point cloud pooling method.

3.1. Motion-guided Sequential Pooling

Current multi-frame detection methods [2,18] mostly explore proposal trajectories to generate high-quality point cloud representations. However, such a scheme relies on frame-by-frame proposal generation, which is not suitable for online detection systems. We observe that in a point cloud sequence, although objects move at different speeds, their motions are relatively smooth. That is to say, we can estimate their motion displacements and roughly localize their positions in preceding frames. To this end, given a point cloud sequence $\{I^t\}_{t=1}^{T}$, we propose to propagate the proposals generated on the current frame I^T to preceding frames $\{I^t\}_{t=1}^{T-1}$ based on their estimated velocities. Since moving objects may slightly deviate from the estimated positions in the preceding frames, we sample the points-of-interest in a cylindrical region of each proposal and gradually increase the diameter of the region by a factor γ as the proposal propagates. Let's denote a proposal of current frame as $(p_x, p_y, p_z, w, l, h, \theta)$, where (p_x, p_y, p_z) denotes its center location, w, l, h and θ denote its width, length, height and yaw angle, respectively. Suppose that the object has a unit-time velocity $\vec{v} = (v_x, v_y)$. The corresponding points-of-interest (x^t, y^t, z^t) sampled from frame t will satisfy the following condition:

$$(x^t - p_x + v_x \cdot \Delta t)^2 + (y^t - p_y + v_y \cdot \Delta t)^2 < (\frac{d^t}{2})^2, \quad (1)$$

where $\Delta t = T - t$ is the time offset of frame t and $d^t = \sqrt{(w^2 + l^2)} \cdot \gamma^{\Delta t + 1}$ is the diameter of cylindrical region.

In our preliminary study, we compare the overall recall rates of foreground points between our method and the trajectory-based method [2]. As can be seen in Tab. 1,

our method can achieve very close result to the the proposal trajectory method on 4-frame sequences. For 8-frame sequences, since it is difficult to estimate the positions of fast-moving objects on distant frames, we use γ=1.1 to recall more points and obtain comparable result to the proposal trajectory method. The recall rates only drop slightly even when 16-frame sequences are used. Interestingly, we find that setting γ=1.1 is not only beneficial for detecting fast-moving objects but also beneficial for improving the performance on slow and stationary objects. We believe this is because the points sampled from regions of different sizes contain multi-level contextual information about the objects, which will be further discussed in Sec. 4.3.

3.2. Region-based Network

Proposal feature encoding. After sampling K points in each proposal, we adopt two encoding schemes to generate proposal features. We first follow [19] to calculate the relative offsets between each point-of-interest $l_i^t \in I^t$ and the nine key-points (eight corner points plus one center point) of the proposal box $\{b_j^t \in P^t : j = 0, ..., 8\}$. The resulted offsets are then converted to spherical coordinates and transformed, via a Multi-layer Perceptron (MLP), to a geometric embedding $g^t \in \mathbb{R}^{K\times D}$ that encodes the spatial correlation between the sampled point and the proposal box. The encoding scheme can be formulated as:

$$g_i^t = \text{MLP}(\mathcal{S}(\{l_i^t - b_j^t\}_{j=0}^8)), \text{for } i = 1, ..., K, \quad (2)$$

where $\mathcal{S} : (x, y, z) \rightarrow (r, \theta, \phi)$ denotes the spherical transform where $r = \sqrt{x^2 + y^2 + z^2}$, $\theta = \arcsin(z/r)$ and $\phi = \arctan(y/x)$ respectively. The second scheme produces a motion embedding $m^t \in \mathbb{R}^{K\times D}$, which encodes the displacements of the points-of-interest at each frame I^t relative to the key-points of the proposal boxes b^0 in the first frame. The time offsets Δt are also concatenated, resulting in the motion embedding at frame t as:

$$m_i^t = \text{MLP}(\text{Concat}(\{l_i^t - b_j^0\}_{j=0}^8, \Delta t)), \text{for } i = 1, ..., K. \quad (3)$$

The proposal feature $f^t \in \mathbb{R}^{K\times D}$ can be formulated as the summation of geometric and motion embeddings:

$$f^t = g^t + m^t. \quad (4)$$

Bidirectional feature aggregation. MSF employs a region-based network to explore *spatial-temporal* dependencies among proposal features and fuse them into global representations for final bounding-box refinement. As shown in Fig. 2, the region-based network is composed of three learning blocks. Each block consists of a traditional Multi-Head Self-Attention (MHSA) layer, followed by a Feed-Forward Network (FFN) with residual connection, and our proposed Bidirectional Feature Aggregation (BiFA) module. The MHSA layer aims to encode rich *spatial* relationships and point dependencies in the proposal for refining point features, while the proposed BiFA module aims to encode *temporal* information by facilitating information exchange between proposals across frames.

Specifically, BiFA involves a forward path and a backward path, representing two ways of information flow along the sequence. Since proposal features have an unordered representation, we leverage the *Max-pool&Repeat* [36] tensor manipulation to obtain a summarized contextual features. In the forward path, aside from the first frame in the sequence, which is concatenated with its own contextual features, each of the other frames is concatenated with the contextual features of its preceding frame along channel dimension. A point-wise convolutional layer is thereby employed to refine the concatenated feature and halve their channels. Given proposal features f^t and f^{t-1} from the current frame and its preceding frame, the forward output of frame t, denote by h_F^t, can be obtained by:

$$h_F^t = \text{Conv}(\text{Concat}(f^t, \text{Repeat} \circ \text{Max-pool}(f^{t-1}))). \quad (5)$$

Unfortunately, introducing the forward path only will lead to information imbalance among different frames, *i.e.*, the current frame receives information from other frames, whereas the last preceding frame receives no information from the sequence. To overcome this limitation, we augment the backward path, where the features of frame t are aggregated with the contextual features of frame $t + 1$, resulting in the backward output h_B^t as follows:

$$h_B^t = \text{Conv}(\text{Concat}(h_F^t, \text{Repeat} \circ \text{Max-pool}(h_F^{t+1}))). \quad (6)$$

In this way, each frame can simultaneously exchange information with its two adjacent frames from both directions, and the information can be propagated to more distant frames after three learning blocks. The resulted proposal features can capture long-term dependencies over the point cloud sequence. It is worthy noting that, the parameters of the convolutional layer in the same path can be shared and all the tensor operations can be performed in parallel. This makes our BiFA module lightweight and much more efficient than other multi-frame fusion modules.

Outputs and loss function. Finally, we aggregate the point-wise features of each proposal into a global representation through a query-based transformer decoder layer, where a learnable query feature $q \in \mathbb{R}^D$ attends to each point vector of proposal features $h^t \in \mathbb{R}^{K\times D}$ through a cross-attention module. The decoder layer can be depicted as follows:

$$\hat{e}^t = \text{Attention}(q, h^t, h^t) + h^t, \quad (7)$$

$$e^t = \text{FFN}(\hat{e}^t) + \hat{e}^t. \quad (8)$$

The decoded outputs across frames $\{e^t\}_{t=0}^T$ are concatenated and passed to a group of detection heads for

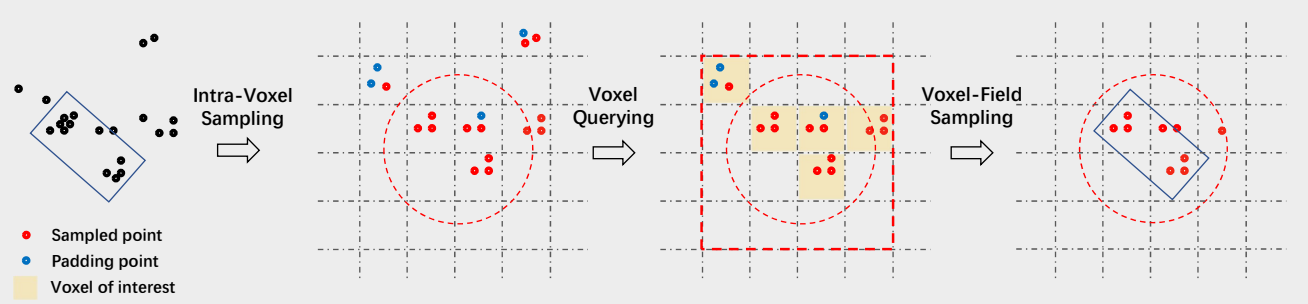

Figure 3. Illustration of our optimized point cloud pooling method. We first perform intra-voxel sampling to keep a fixed number of points in each voxel. Then we query $n \times n$ voxels fields for each proposal and uniformly draw points from the non-empty voxels within.

Table 2. The latency of point cloud pooling on 1-frame, 4-frames and 8-frames sequences.

	N=168k	*N*=674k	*N*=1382k
Cylindrical Pooling	8.2ms	25.2ms	40.1ms
Our Optimized	2.3ms	3.4ms	5.0ms

bounding-box refinement. The overall loss function $\mathcal{L}_{\text{total}}$ is the summation of the confidence prediction loss $\mathcal{L}_{\text{conf}}$ and the box regression loss $\mathcal{L}_{\text{reg}}$ as follows:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{conf}} + \alpha \mathcal{L}_{\text{reg}}, \tag{9}$$

where α is a balancing hyper-parameter. We adopt the same binary cross entropy loss and box regression loss employed in CT3D [19] as our $\mathcal{L}_{\text{conf}}$ and $\mathcal{L}_{\text{reg}}$.

3.3. Efficient Point Cloud Pooling

In this subsection, we optimize the point cloud pooling method to sample a fixed number of points more efficiently from the cylindrical region of each proposal. As shown in Fig. 3, we perform proposal-based point sampling in two steps, *i.e.*, the intra-voxel sampling and the voxel-field sampling. In the first step, the input space is discretized into a voxel grid with voxel size v, and each point corresponds to a voxel coordinates $(\lfloor \frac{x}{v} \rfloor, \lfloor \frac{y}{v} \rfloor)$. Here the voxel partition in z-axis is omitted considering that the cylindrical regions have unlimited height. Then, up to k points are kept in each voxel and padding points are inserted if the voxel has less than k points. Due to the high memory consumption required to store all voxels in a dense grid, we follow [14] to store only non-empty voxels in a contiguous memory and use a hash table to store the mappings between the coordinates of non-empty voxels and their indices in memory space.

In the second step, we first query n-by-n voxel-field for each proposal and calculate the coordinates of the voxels within. Next, we convert the coordinates of voxel into hashed keys and look up the table for querying the sampled points generated from the first hierarchy. The hash-table will return "-1" if no keys are found, which means it is an empty voxel. The queried points are then drawn from these voxels and stored in an output buffer if the following conditions are met: 1) it is a valid point and 2) it falls into the cylindrical region, *i.e.* satisfying Eq. 1.

Complexity and efficiency We perform an in-depth analysis of our optimized pooling method and the previous cylindrical pooling methods [2, 19, 30] in continuous space. Given N points, M proposals and the requirement of sampling K points in each proposal, the original method costs $\mathcal{O}(NM) + \mathcal{O}(MK)$ complexity due to the calculation of pair-wise distances between points and proposals. In contrast, our optimized version costs $\mathcal{O}(N)+\mathcal{O}(M)+\mathcal{O}(MK)$ complexity, where the first term is the cost of intra-voxel sampling, the second term is the cost of querying voxel-field for each proposal and the third term is the cost of drawing points from the queried voxels. We evaluate the latency of point cloud pooling on sequences with different lengths. As shown in Tab. 2, our optimized pooling method can achieve an 8× speedup over the original pooling method.

4. Experiment

4.1. Dataset and Implementation Details

Dataset. We evaluate our MSF on Waymo Open Dataset (WOD), which is a large-scale multi-modality dataset for autonomous driving. WOD contains 1,150 sequences, which are divided into 798 training, 202 validation, and 150 testing sequences, respectively. Each sequence is 20s long, captured by a 64-line LiDAR sensor at 10Hz frequency. The evaluation metrics used in WOD are mean average precision (mAP) and mAP weighted by heading accuracy (mAPH). Three object classes, "Vehicle", "Pedestrian" and "Cyclist", are evaluated. Each object class is further categorized into two levels of difficulties, LEVEL1 and LEVEL2. The former refers to objects with more than 5 points and the latter refers to objects with less than 5 but at least 1 point.

Implementation. We employ the traditional CenterPoint [28, 33, 36] as our region proposal network (RPN). To include motion information, we concatenate four adjacent frames at input and add one additional head for pre-

Table 3. Performance comparison on the validation set of Waymo Open Dataset.

Method	Frames	ALL (3D mAPH)		Vehicle (AP/APH)		Pedestrian (AP/APH)		Cyclist (AP/APH)	
		L1	L2	L1	L2	L1	L2	L1	L2
SECOND [28]	1	63.05	57.23	72.27/71.69	63.85/63.33	68.70/58.18	60.72/51.31	60.62/59.28	58.34/57.05
PointPillar [9]	1	63.33	57.53	71.60/71.00	63.10/62.50	70.60/56.70	62.90/50.20	64.40/62.30	61.90/59.90
IA-SSD [35]	1	64.48	58.08	70.53/69.67	61.55/60.80	69.38/58.47	60.30/50.73	67.67/65.30	64.98/62.71
LiDAR R-CNN [10]	1	66.20	60.10	73.50/73.00	64.70/64.20	71.20/58.70	63.10/51.70	68.60/66.90	66.10/64.40
RSN [26]	1	-	-	75.10/74.60	66.00/65.50	77.80/72.70	68.30/63.70	-	-
PV-RCNN [21]	1	69.63	63.33	77.51/76.89	68.98/68.41	75.01/65.65	66.04/57.61	67.81/66.35	65.39/63.98
Part-A2 [24]	1	70.25	63.84	77.05/76.51	68.47/67.97	75.24/66.87	66.18/58.62	68.60/67.36	66.13/64.93
Centerpoint [33]	1	-	65.50	-	-/66.20	-	-/62.60	-	-/67.60
VoTR [14]	1	-	-	74.95/74.25	65.91/65.29	-	-	-	-
VoxSeT [6]	1	72.24	66.22	74.50/74.03	65.99/65.56	80.03/72.42	72.45/65.39	71.56/70.29	68.95/67.73
SST-1f [3]	1	-	-	76.22/75.79	68.04/67.64	81.39/74.05	72.82/65.93	-	-
SWFormer-1f [25]	1	-	-	77.8/77.3	69.2/68.8	80.9/72.7	72.5/64.9	-	-
PillarNet [20]	1	74.60	68.43	79.09/78.59	70.92/70.46	80.59/74.01	72.28/66.17	72.29/71.21	69.72/68.67
PV-RCNN++ [22]	1	75.21	68.61	79.10/78.63	70.34/69.91	80.62/74.62	71.86/66.30	73.49/72.38	70.70/69.62
3D-MAN [31]	16	-	-	74.53/74.03	67.61/67.14	71.7/67.7	62.6/59.0	-	-
SST-3f [3]	3	-	-	78.66/78.21	69.98/69.57	83.81/80.14	75.94/72.37	-	-
SWFormer-3f [25]	3	-	-	79.4/78.9	71.1/70.6	82.9/79.0	74.8/71.1	-	-
CenterFormer [37]	4	77.0	73.2	78.1/77.6	73.4/72.9	81.7/78.6	77.2/74.2	75.6/74.8	73.4/72.6
CenterFormer [37]	8	77.3	73.7	78.8/78.3	74.3/73.8	82.1/79.3	77.8/75.0	75.2/74.4	73.2/72.3
MPPNet [2]	4	79.83	74.22	81.54/81.06	74.07/73.61	84.56/81.94	77.20/74.67	77.15/76.50	75.01/74.38
MPPNet [2]	16	80.40	74.85	82.74/**82.28**	75.41/74.96	84.69/82.25	77.43/75.06	77.28/76.66	75.13/74.52
MSF (ours)	4	80.20	74.62	81.36/80.87	73.81/73.35	85.05/82.10	77.92/75.11	78.40/77.61	76.17/75.40
MSF (ours)	8	**80.65**	**75.46**	**82.83**/82.01	**75.76/75.31**	**85.24/82.21**	**78.32/75.61**	**78.52/77.74**	**76.32/75.47**

dicting the velocity of objects. In our experiments, we first train RPN using the official settings of OpenPCDet[1], and use it to generate proposals of WOD. Based on these proposals, we then train our region-based network for 6 epochs by using the ADAM optimizer with an initial learning rate of 0.003 and a batch size of 16. The learning rate is decayed with One-Cycle policy and the momentum is set between [85%, 95%]. During the training, we sample region proposals with IoU>0.5 and conduct proposal augmentation following PointRCNN [23]. A number of 128 raw LiDAR points are randomly sampled for each proposal. The voxel size v and the points-per-voxel k used in voxel-based sampling are set to 0.4 and 32, respectively. The feature dimension of each point in the learning block is set to 256. At the training stage, we use the intermediate supervision by adding loss to the output of each learning block and sum all the intermediate losses to train the model. At the test stage, we only use the bounding boxes and the confidence scores predicted from the last learning block.

4.2. Results on WOD

Waymo Validation Set. Tab. 3 compares the performance of MSF and the current state-of-the-art methods. As can be seen, multi-frame detectors generally outperform single-frame methods. The best two multi-frame methods by far are MPPNet [2] based on proposal trajectory and CenterFormer [37] based on transformer model. Using the same number of frames, our MSF method significantly outperforms CenterFormer by 3.4% APH and 1.2% APH on LEVEL1 and LEVEL2, respectively. This demonstrates that fusing multiple frame feature at the region level is more effective than the fusion with convolutional features. Compared with MPPNet [2], which is also based on region-level fusion, our method outperforms it on almost all cases, except for the APH of Vehicle LEVEL1. It should be noted that our MSF method only use 8 frames, while MPPNet uses 16 frames. MSF uses different pooling sizes in different frames, therefore generating multi-level contextual features for each object. Specifically, MSF models with 4- and 8-frames achieve the mAPH of 74.62% and 75.13%, respectively, recording new state-of-the-arts. In addition, both CenterFormer and MPPNet extract features on each frame of the sequence with the need of a memory bank to store the intermediate results of preceding frames, while our method performs proposal generation only on the current frame, which is more practical to online inference.

Waymo Test Set. In Tab. 4, we show the results of our 8-frame model by submitting the prediction result to the online server for evaluation. Our method outperforms all the previously published methods. In particular, the improvements on Vehicle and Pedestrian classes are significant, outperforming the best competitor, *i.e.*, CenterFormer, by 1.91% APH and 1.89% APH on LEVEL2, respectively.

[1] https://github.com/open-mmlab/OpenPCDet/

Table 4. Performance comparison on the test set of Waymo Open Dataset.

Method	ALL (3D mAPH)		Vehicle (AP/APH)		Pedestrian (AP/APH)		Cyclist (AP/APH)	
	L1	L2	L1	L2	L1	L2	L1	L2
PointPillar [9]	-	-	68.10	60.10	68.00/55.50	61.40/50.10	-	-
StarNet [15]	-	-	61.00	54.50	67.80/59.90	61.10/54.00	-	-
M3DETR [5]	67.1	61.9	77.7/77.1	70.5/70.0	68.2/58.5	60.6/52.0	67.3/65.7	65.3/63.8
3D-MAN [31]	-	-	78.28	69.98	69.97/65.98	63.98/60.26	-	-
PV-RCNN++ [22]	75.7	70.2	81.6/81.2	73.9/73.5	80.4/75.0	74.1/69.0	71.9/70.8	69.3/68.2
CenterPoint [33]	77.2	71.9	81.1/80.6	73.4/73.0	80.5/77.3	74.6/71.5	74.6/73.7	72.2/71.3
RSN [26]	-	-	80.30	71.60	78.90/75.60	70.70/67.80	-	-
SST-3f [3]	78.3	72.8	81.0/80.6	73.1/72.7	83.3/79.7	76.9/73.5	75.7/74.6	73.2/72.2
MPPNet [2]	80.59	75.67	84.27/83.88	77.29/76.91	84.12/81.52	78.44/75.93	77.11/76.36	74.91/74.18
CenterFormer [37]	80.91	76.29	85.36/84.94	78.68/78.28	85.22/ 82.48	80.09/77.42	76.21/75.32	74.04/73.17
MSF (ours)	**81.74**	**76.96**	**86.07/85.67**	**79.20/78.82**	**85.99/83.10**	**80.61/77.82**	**77.29/76.44**	**75.09/74.25**

Table 5. Ablation experiments on the WOD validation set. "ME" refers to using Motion Embedding and "SA" refers to using Self-Attention modules. "Q" means using query-based decoder layer to generate global representation of proposal features and "M" means using single max-pooling layer for global feature generation. APH scores on LEVEL1 and LEVEL2 are reported.

ME	SA	BiFA	Decoder	L1	L2
✓	✓	✓	Q	80.20	74.62
✘	✓	✓	Q	80.03 (-0.17)	74.50 (-0.12)
✓	✘	✓	Q	76.51 (-3.69)	71.91 (-2.71)
✓	✓	✘	Q	78.25 (-1.95)	73.11 (-1.51)
✓	✓	✓	M	79.54 (-0.66)	74.08 (-0.54)

Table 6. Bidirectional feature aggregation vs. unidirectional feature aggregation.

Forward	Backward	L1	L2
✓	✓	80.20	74.62
✓	✘	79.64 (-0.56)	74.35 (-0.27)
✘	✓	78.97 (-1.23)	73.77 (-0.85)

Table 7. The performance by integrating the proxy points and MLP mixer from MPPNet. "SA" and "Mixer" denote the self-attention and MLP Mixer [2], respectively. The APH scores of LEVEL2 on three object classes are reported.

Config	Vehicle.	Pedestrian.	Cyclist
Raw + SA	73.35	75.11	75.40
Proxy + SA	73.12 (-0.23)	74.20 (-0.91)	74.32 (-1.08)
Proxy + Mixer	73.45(+0.10)	74.13 (-0.98)	74.39 (-1.01)

4.3. Ablation Studies

In this section, we conduct in-depth analysis on MSF by evaluating the effectiveness of each individual component of it. We report the APH metric of our 4-frame model on the WOD validation set. The ablation study results are shown in Tab. 5 to Tab. 8, respectively.

Motion embedding. As can be seen from the 1st and 2nd rows of Tab. 5, without motion encoding, the performance of our proposed MSF will drop by 0.17% on LEVEL1 and 0.12% in LEVEL2. This indicates that the motion information is beneficial to infer the object's geometry information in point cloud sequences.

Self-attention. As can be seen from the 1st and 3rd rows of Tab. 5, without using self-attention for spatial interactions, MSF will suffer from significant performance drop, 3.69% on LEVEL1 and 2.71% on LEVEL2. This indicates that the intra-frame interaction is vital for learning internal geometry information of the proposals.

Bidirectional feature aggregation. From the 1st and 4th rows of Tab. 5, we can find that enforcing temporal interactions among hidden features of proposals by using the proposed BiFA module can bring an improvement of 1.95% on LEVEL1 and 1.51% on LEVEL2, which demonstrates the benefits of exploring long-term dependencies in the point cloud sequence. We also conduct experiments by performing unidirectional feature aggregation with either forward or backward path. The results are shown in Tab. 6, from which we see that the unidirectional model obtains lower APH scores than that of the bidirectional model.

Query-based decoder layer. As shown in the last row of Tab. 5, by replacing the final query-based decoder layer with a single max-pooling layer, the performance will drop slightly by 0.66% and 0.54%. This is because the final proposal representation after decoder can be regarded as a weighted sum of all point features, and the decoding weights of different points can intrinsically introduce more dynamics for feature representation.

Spatial modeling with proxy points and MLP mixer. We perform an analysis by integrating the proxy points and MLP mixer developed in our best competitor MPPNet [2] into MSP. We first follow MPPNet to generate proxy point

Table 8. The APH (L2) scores on objects with different velocities using the 8-frame model. The objects are categorized into stationary (<0.2m/s), slow (0.2∼1m/s), medium (1∼6m/s) and fast (>6m/s) groups.

	Stationary	Slow	Medium	Fast
γ=1.0	70.32	70.30	75.60	80.32
γ=1.1	70.43	71.17	77.56	82.48
γ=1.2	70.35	71.23	77.44	82.31

of each proposal and apply a set abstraction [17] to aggregate the point-wise features to every proxy point. As shown in the 1st and 2nd rows of Tab. 7, using features of proxy points will degrade the performance on Vehicle and more significantly on Pedestrian and Cyclist classes. After replacing the self-attention module with the MLP mixer, the performance degradation on Pedestrian and Cyclist classes still exists, as shown in the 3rd row of Tab. 7. This phenomenon is contradictory to the conclusion in MPPNet that using proxy points to formulate proposal features is more effective than using raw points. We believe this is because MPPNet applies per-frame detection, and thus the proposals across frames may have different sizes, while proxy points can provide consistent representations for different frames. In contrast, our method uses propagated proposals with the same size over the sequence, and hence our proposal features are naturally on the same scale. For small objects such as pedestrian and cyclist, the features from raw points can provide more fine-grained details than proxy points.

Effects of γ. We evaluate how γ affects the performance on the objects with different speeds. As shown in Tab. 8, gradually expanding the proposal region with γ=1.1 will bring substantial improvements on the moving objects. This demonstrates that our model is able to capture the information of fast moving objects, even in the distant frames. Interestingly, we also see improvements on slow and stationary objects. This is because the different pooling sizes could provide multi-level contextual information, which is beneficial for detecting the objects. As shown in the 3rd row of Tab. 8, no further improvements can be made by increasing the value of γ to 1.2.

4.4. Runtime Analysis

Fig. 4 illustrates the latency decomposition of different methods. Here the latency is computed as the average runtime over 100 samples that are randomly drawn from WOD validation set, which is measured by a single Nvidia GeForce RTX A6000 GPU. Since all the methods employ the same backbone network, we exclude the runtime of the backbone from the latency for better comparison. For MPPNet and CenterFormer, we assume that the features from the preceding frames are already available in the memory bank. As can be seen, MPPNet costs much time in the feature encoding stage because it performs set-abstraction [17] to group raw points on the proxy points, which requires manipulating point clouds in continuous space. In comparison, our method takes only 4 milliseconds to encode raw points as proposal features. The overall latency of MPPNet, CenterFormer and MSF are 99, 80 and 50 milliseconds, respectively. Tab. 9 further decomposes the network latency regarding the spatial and temporal modules in MPPNet and MSF. As can be seen, the self-attention module and BiFA module are much more efficient than MLP-Mixer and cross-attention module, respectively. Thanks to our optimized point cloud pooling, MSF achieves higher efficiency even compared with CenterFormer, which performs feature fusion directly on the convolutional features.

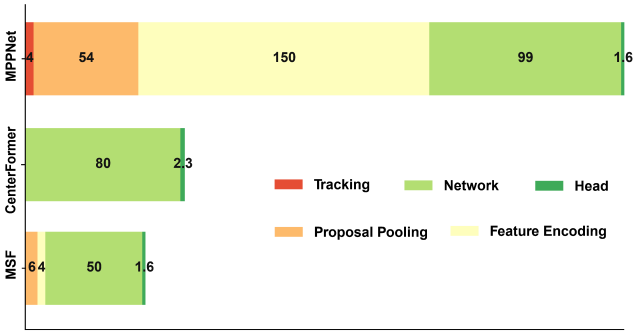

Figure 4. Comparison of the runtime of different methods.

Table 9. The runtime decomposition of MPPNet and MSF.

MPPNet		MSF	
MLP-Mixer	Cross-Attention	Self-Attention	BiFA
75 ms	24 ms	36 ms	12 ms

5. Conclusion

We presented a novel motion-guided sequential fusion method, namely MSF, for 3D object detection from point cloud sequence. Unlike previous multi-frame detectors that performed feature extraction on each frame of the sequence, MSF adopted a proposal propagation strategy to mine points-of-interest based on the proposals generated on the current frame, therefore reducing the redundant computations and relieving reliance on the preceding results. A bidirectional feature aggregation module was proposed to enable cross-frame interaction between proposal features, facilitating MSF to capture long-term dependencies over the sequence. Besides, we optimized the point cloud pooling process, allowing MSF to process large-scale point cloud sequences in milliseconds. The proposed MSF achieved state-of-the-art performance on the Waymo Open Datasets and it was more efficient than other multi-frame detectors. In future research, we plan to extend MSF to generate detection priors on the future frames to further reduce the overall computation of multi-frame detection.

References

[1] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1907–1915, 2017. 1

[2] Xuesong Chen, Shaoshuai Shi, Benjin Zhu, Ka Chun Cheung, Hang Xu, and Hongsheng Li. Mppnet: Multi-frame feature intertwining with proxy points for 3d temporal object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2022. 1, 2, 3, 5, 6, 7

[3] Lue Fan, Ziqi Pang, Tianyuan Zhang, Yu-Xiong Wang, Hang Zhao, Feng Wang, Naiyan Wang, and Zhaoxiang Zhang. Embracing single stride 3d object detector with sparse transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8458–8468, June 2022. 2, 6, 7

[4] Lue Fan, Xuan Xiong, Feng Wang, Naiyan Wang, and ZhaoXiang Zhang. Rangedet: In defense of range view for lidar-based 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2918–2927, October 2021. 2

[5] Tianrui Guan, Jun Wang, Shiyi Lan, Rohan Chandra, Zuxuan Wu, Larry Davis, and Dinesh Manocha. M3detr: Multi-representation, multi-scale, mutual-relation 3d object detection with transformers. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (CVPR)*, pages 772–782, 2022. 7

[6] Chenhang He, Ruihuang Li, Shuai Li, and Lei Zhang. Voxel set transformer: A set-to-set approach to 3d object detection from point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8417–8427, June 2022. 1, 6

[7] Chenhang He, Hui Zeng, Jianqiang Huang, Xian-Sheng Hua, and Lei Zhang. Structure aware single-stage 3d object detection from point cloud. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11873–11882, 2020. 1, 2

[8] Rui Huang, Wanyue Zhang, Abhijit Kundu, Caroline Pantofaru, David A Ross, Thomas Funkhouser, and Alireza Fathi. An lstm approach to temporal 3d object detection in lidar point clouds. In *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2020. 1, 2

[9] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 12697–12705, 2019. 1, 2, 6, 7

[10] Zhichao Li, Feng Wang, and Naiyan Wang. Lidar r-cnn: An efficient and universal 3d object detector. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 6

[11] Zhidong Liang, Zehan Zhang, Ming Zhang, Xian Zhao, and Shiliang Pu. Rangeioudet: Range image based real-time 3d object detector optimized by intersection over union. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7140–7149, June 2021. 2

[12] Zhijian Liu, Haotian Tang, Yujun Lin, and Song Han. Point-voxel cnn for efficient 3d deep learning. *Advances in Neural Information Processing Systems*, 32, 2019. 2

[13] Wenjie Luo, Bin Yang, and Raquel Urtasun. Fast and furious: Real time end-to-end 3d detection, tracking and motion forecasting with a single convolutional net. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3569–3577, 2018. 2

[14] Jiageng Mao, Yujing Xue, Minzhe Niu, Haoyue Bai, Jiashi Feng, Xiaodan Liang, Hang Xu, and Chunjing Xu. Voxel transformer for 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3164–3173, October 2021. 1, 5, 6

[15] Jiquan Ngiam, Benjamin Caine, Wei Han, Brandon Yang, Yuning Chai, Pei Sun, Yin Zhou, Xi Yi, Ouais Alsharif, Patrick Nguyen, et al. Starnet: Targeted computation for object detection in point clouds. *arXiv preprint arXiv:1908.11069*, 2019. 7

[16] Charles R. Qi, Or Litany, Kaiming He, and Leonidas J. Guibas. Deep hough voting for 3d object detection in point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019. 2

[17] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in neural information processing systems*, pages 5099–5108, 2017. 2, 8

[18] Charles R. Qi, Yin Zhou, Mahyar Najibi, Pei Sun, Khoa Vo, Boyang Deng, and Dragomir Anguelov. Offboard 3d object detection from point cloud sequences. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6134–6144, June 2021. 1, 2, 3

[19] Hualian Sheng, Sijia Cai, Yuan Liu, Bing Deng, Jianqiang Huang, Xian-Sheng Hua, and Min-Jian Zhao. Improving 3d object detection with channel-wise transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2743–2752, October 2021. 2, 4, 5

[20] Guangsheng Shi, Ruifeng Li, and Chao Ma. Pillarnet: High-performance pillar-based 3d object detection. 2022. 6

[21] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 1, 2, 6

[22] Shaoshuai Shi, Li Jiang, Jiajun Deng, Zhe Wang, Chaoxu Guo, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn++: Point-voxel feature set abstraction with local vector representation for 3d object detection. *arXiv preprint arXiv:2102.00463*, 2021. 6, 7

[23] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–779, 2019. 2, 6

[24] Shaoshuai Shi, Zhe Wang, Xiaogang Wang, and Hongsheng Li. Part-aˆ 2 net: 3d part-aware and aggregation neural net-

work for object detection from point cloud. *arXiv preprint arXiv:1907.03670*, 2019. 6

[25] Pei Sun, Mingxing Tan, Weiyue Wang, Chenxi Liu, Fei Xia, Zhaoqi Leng, and Dragomir Anguelov. Swformer: Sparse window transformer for 3d object detection in point clouds. 2022. 6

[26] Pei Sun, Weiyue Wang, Yuning Chai, Gamaleldin Elsayed, Alex Bewley, Xiao Zhang, Cristian Sminchisescu, and Dragomir Anguelov. Rsn: Range sparse net for efficient, accurate lidar 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR*, pages 5725–5734, 2021. 6, 7

[27] Dominic Zeng Wang and Ingmar Posner. Voting for voting in online point cloud object detection. *Robotics: Science and Systems*, 1(3):10–15607, 2015. 1

[28] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 18(10):3337, 2018. 1, 2, 5, 6

[29] Zetong Yang, Yanan Sun, Shu Liu, and Jiaya Jia. 3dssd: Point-based 3d single stage object detector. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 1, 2

[30] Zetong Yang, Yanan Sun, Shu Liu, Xiaoyong Shen, and Jiaya Jia. STD: sparse-to-dense 3d object detector for point cloud. In *Proceedings of the IEEE international conference on computer vision (ICCV)*, 2019. 2, 5

[31] Zetong Yang, Yin Zhou, Zhifeng Chen, and Jiquan Ngiam. 3d-man: 3d multi-frame attention network for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1863–1872, June 2021. 1, 2, 6, 7

[32] Junbo Yin, Jianbing Shen, Chenye Guan, Dingfu Zhou, and Ruigang Yang. Lidar-based online 3d video object detection with graph-based message passing and spatiotemporal transformer attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 1, 2

[33] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. Center-based 3d object detection and tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11784–11793, June 2021. 2, 3, 5, 6, 7

[34] Yifan Zhang, Qingyong Hu, Guoquan Xu, Yanxin Ma, Jianwei Wan, and Yulan Guo. Not all points are equal: Learning highly efficient point-based detectors for 3d lidar point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18953–18962, June 2022. 2

[35] Yifan Zhang, Qingyong Hu, Guoquan Xu, Yanxin Ma, Jianwei Wan, and Yulan Guo. Not all points are equal: Learning highly efficient point-based detectors for 3d lidar point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18953–18962, 2022. 6

[36] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4490–4499, 2018. 1, 2, 4, 5

[37] Zixiang Zhou, Xiangchen Zhao, Yu Wang, Panqu Wang, and Hassan Foroosh. Centerformer: Center-based transformer for 3d object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2022. 1, 6, 7