MAD: Memory-Augmented Detection of 3D Objects

Ben Agro[†] Sergio Casas[†] Patrick Wang[†] Thomas Gilles
Raquel Urtasun
{tgilles, urtasun}@waabi.ai

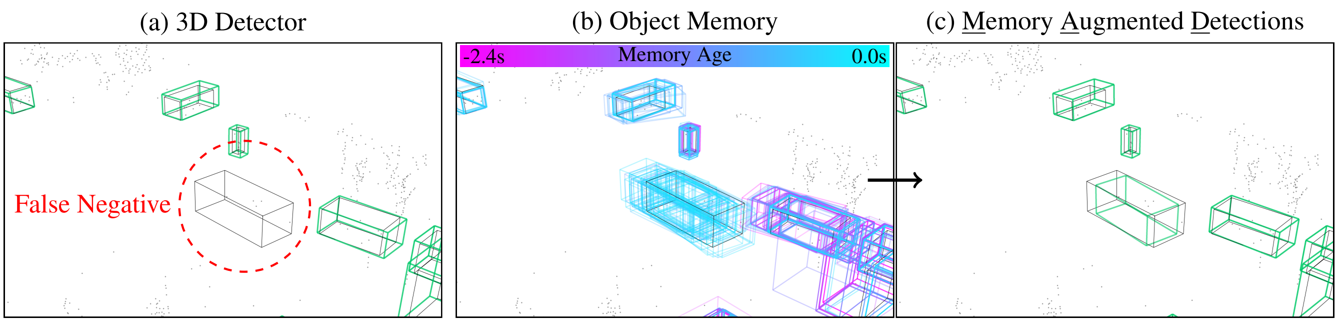

Figure 1. Detectors without long-term temporal fusion (a) miss heavily occluded objects. Our approach enhances detectors (b) to remember past predictions, (c) recovering from occlusion. Detections are in green, labels are in black, lidar points are in gray.

Abstract

To perceive, humans use memory to fill in gaps caused by our limited visibility, whether due to occlusion or our narrow field of view. However, most 3D object detectors are limited to using sensor evidence from a short temporal window (0.1s-0.3s). In this work, we present a simple and effective add-on for enhancing any existing 3D object detector with long-term memory regardless of its sensor modality (e.g., LiDAR, camera) and network architecture. We propose a model to effectively align and fuse object proposals from a detector with object proposals from a memory bank of past predictions, exploiting trajectory forecasts to align proposals across time. We propose a novel schedule to train our model on temporal data that balances data diversity and the gap between training and inference. By applying our method to existing LiDAR and camera-based detectors on the Waymo Open Dataset (WOD) and Argoverse 2 Sensor (AV2) dataset, we demonstrate significant improvements in detection performance (+2.5 to +7.6 AP points). Our method attains the best performance on the WOD 3D detection leaderboard among online methods (excluding ensembles or test-time augmentation).

1 Introduction

Most self-driving vehicles (SDVs) utilize a 3D object detector to recognize and localize objects in 3D space. This task is challenging due to occlusion, large intra-class variability, and distant objects, which typically have limited sensor observations. To overcome these challenges, human drivers rely on their memory. For example, they may drive more cautiously when remembering a previously observed but now occluded cyclist, who may suddenly enter the road.

A common approach for improving 3D object detectors is to aggregate a short temporal window of past sensor observations. Towards this goal, most existing LiDAR-based methods transform a short buffer of sensor data into the current ego (SDV) coordinate frame to align past and current evidence [1, 39, 58, 74, 75]. Similarly, camera-based methods stack multiple images [46, 78] as input to existing architectures. These methods cannot handle long temporal sequences due to computational and memory constraints. Moreover, temporal stacks of 3D/Bird's-Eye-View (BEV) representations like point clouds or lifted camera features require a large receptive field, especially for fast-moving objects [30], further increasing computational burden.

There is a growing interest in long-term temporal fusion. Scene-level memory approaches [14, 17, 31] recurrently fuse scene-level features, but they can struggle to capture relevant foreground objects. Other approaches associate objects in memory over time via tracking [8, 20, 30, 32], aggregating past information for each particular object. However, the associations from the tracker may contain mistakes that can compound over time and lead to information loss. Other methods leverage attention from current detection proposals to the past sensor or object information [15, 18, 74]. Still, they can be challenging to scale to long histories and suffer from false negatives as the proposals refined into the final detections only come from the

[†]Work done while at Waabi

present time.

In this paper, we present a simple and sensor-agnostic add-on for enhancing any existing 3D object detector with long-term memory. We refer to it as MAD — short for Memory-Augmented Detection, and Fig. 1 illustrates the high-level idea. MAD is a transformer-based model that fuses proposals from a detector with proposals from a memory bank representing past beliefs. Inspired by recent developments [4], we exploit joint detection and trajectory forecasting. By storing explicit trajectory forecasts in the memory bank, we can estimate object poses at arbitrary future timestamps for all the objects in the memory. This enables us to enrich the set of proposals by aligning memory proposals with the current observations.

Training with temporal data can be challenging: backpropogation through consecutive training examples consumes prohibitive amounts of memory, training on long sequences can cause over-fitting when back-propagating on every example, and using memory warm-up can slow down training. We design a more effective training schedule that begins with short temporal sequences and progressively increases the length, exploiting high data diversity early and closing the gap with inference towards the end. To ensure the model learns to trust the memory when training on short sequences, we use cached model outputs from previous training iterations.

We demonstrate the generality of our approach by enhancing existing LiDAR-based and camera-based 3D object detection networks with MAD, and show considerable improvements over the base detectors on two large-scale datasets: Waymo Open dataset (WOD) [57] and Argoverse 2 Sensor dataset (AV2) [68]. Notably, SAFDNet [80] enhanced with MAD achieves state-of-the-art performance on WOD for online detection methods without requiring ensembles or test-time augmentation.

2 Related Work

3D Object Detection: We can categorize 3D detectors by their input modality (e.g., LiDAR, camera), scene representation (e.g., point clouds, voxels), and number of stages (e.g., single-stage or multi-stage)

LiDAR-based methods commonly represent the input as voxels [12, 27, 56, 71, 80, 81], pillars [23, 25, 55, 61, 66, 72], or point clouds [43, 53, 54, 69, 73]. A widely used approach for including temporal LiDAR information is *point aggregation*, which involves transforming past point clouds into a common coordinate frame and processing the aggregated point cloud. These approaches are usually limited to ≤ 5 past LiDAR frames due to computational constraints [30] in online applications like autonomous driving. Another drawback is that point aggregation does not align moving objects, requiring a larger receptive field in the detector backbone the longer the temporal horizon is [30, 80].

Camera-based 3D detection is challenging because of missing depth information. One approach is to produce 3D bounding boxes from image features by estimating depth, 3D size, and orientation [7, 41, 64, 70]. Other methods leverage voxel [47, 48, 51] or point cloud [65, 76] representations by predicting pixel depth distributions to lift 2D features to 3D. Stacking and processing past camera images or feature maps is a common but expensive method for temporal fusion [2, 19, 46, 52, 78].

Regardless of modality, we can further categorize 3D detectors as single-stage or multi-stage. Single-stage methods produce detections from sensor data [6, 12, 64, 71, 80, 81] with a single deep neural network. Multi-stage methods use bounding box proposals from a first stage (or randomly initialized proposals) to gather features (e.g., with RoIPool [49], RoIAlign [16], interpolations [75], or attention [4, 28, 33–35, 67, 82]) and iteratively refine the bounding boxes.

MAD is a sensor-modality-agnostic module that we can add to any detector as a subsequent refinement stage. In our work we utilize CenterPoint [75], SAFDNet [80], HEDNet [81], FCOS3D [64], and BEVMap [6] as proposal networks.

Long-Term Temporal Fusion for 3D Detection: Various works attempt to solve the shortcomings of sensor aggregation by learning to use multiple seconds of sensor evidence to improve object detection.

In this line of work, the *scene-based* paradigm uses recurrent fusion of scene-level features [14, 17, 31, 75], with some methods relying on multiple traversals of the scene [77]. A challenge of this approach is focusing and aligning features from relevant and dynamic foreground objects, which past works addressed by transforming feature maps, using segmentation to focus on foreground objects [17], and using deformable attention or convolution to align features of moving objects [22]. Processing both foreground and background areas can be computationally expensive. It is worth noting that many of these methods are single-stage detectors [14, 17] and which we could use as a detection proposal network with MAD.

Alternatively, the *object-based* paradigm focuses on the foreground by using detection proposals. *Detect-track-fuse* methods are a sub-family of object-based methods that associate previous detections over time to create tracks, and these tracks summarize information from past sensor evidence [8, 20, 30, 30, 32, 79]. However, in complex situations like pedestrian crowds, association over time can be difficult due to heavy occlusions and erratic behavior, potentially leading to false negatives or identity switches in the tracks. *Object-to-scene* approaches mitigate the shortcomings of association by directly using current detection proposals to aggregate historical scene-level information using hand-crafted feature aggregation modules [15]

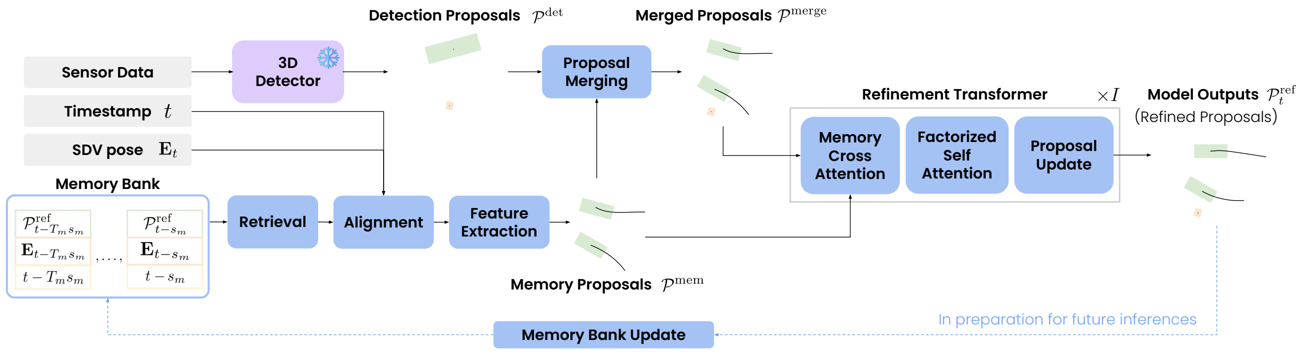

Figure 2. MAD is a plug-and-play module that enhances any off-the-shelf 3D detector (kept frozen) with long-term memory.

or attention mechanisms [82]. These approaches can be difficult to scale to long history horizons as they require re-processing past sensor evidence or dense feature maps based on the current proposals (e.g., [15, 82] only use 0.7s of history). Finally, *object-to-object* methods use past object detections to improve current object detections without explicit tracking, e.g., by cross-attending from current detection proposals to past detections [63, 74], or using hand-coded attention matrices based on distance [18]. Overall, most object-based methods share some deficiencies: Many only refine detection proposals produced by current sensor evidence and struggle to recover from missing proposals [8, 15, 18, 20, 74]. Others naively concatenate past detections with current proposals [63], which can lead to alignment issues for dynamic objects and miss-calibration in the proposal scores, as the model should trust historical proposals less than current proposals.

Our proposed method, MAD, performs object-based temporal fusion without requiring explicit object association, aligns the memory in space and time by with trajectory forecasting, and can recover from missing proposals by using and rescoring proposals from the memory bank.

3 Memory Augmented 3D Object Detection

3D object detectors take a short temporal window of sensor data as input and produce a set of detections. Existing approaches typically struggle to perceive occluded and distant objects with limited sensor observations. To tackle these challenges, we propose MAD, a plug-and-play module to enhance existing 3D object detectors with the ability to perform long-horizon temporal fusion. Our only requirement from the detector is that each detection includes an object bounding box, multi-class confidence scores, and a feature vector capturing local context. We demonstrate the generality of MAD by augmenting and improving various LiDAR-based and camera-based detectors.

We enable long-horizon temporal understanding through a memory bank that captures all the relevant information on objects, including where we expect them to move. These trajectory forecasts allow us to align the memory objects with the current detector proposals in space and time. Importantly, we do not require the object detector to provide motion forecasts; instead, MAD computes them. To compensate for ego-motion, we assume the ego is localized — which is the norm in modern self-driving platforms [57, 68] — and store the ego pose in the memory along with the model outputs.

3.1. Model

We start with an overview of our model; refer to Figure 2 for an illustration. At every inference step, MAD takes as input the *detection proposals*, the current timestamp t (e.g., LiDAR sweep-end time or camera capture time), and the ego pose $\mathbf{E}_t$ in a global coordinate frame. It then retrieves objects from memory, aligns them spatially with $\mathbf{E}_t$ and temporally to t, and extracts high-dimensional features from the aligned boxes and trajectory forecasts. We refer to the aligned boxes and trajectory forecasts with the extracted features as *memory proposals*. A proposal merging mechanism then fuses detection and memory proposals by rescoring their confidence scores and applying standard post-processing. Finally, our refinement transformer iterative refines the object detections and trajectory forecasts in the *merged proposals* with cross-attention to the memory and factorized self-attention. In preparation for future inferences, the memory bank is then updated by appending the model outputs (a.k.a. *refined proposals*) and removing older model outputs to keep the memory bounded in size.

Proposal representation: We define object *proposals* $\mathcal{P} = (\mathbf{B}, \mathbf{C}, \mathbf{T}, \mathbf{Q})$ with N bounding boxes $\mathbf{B} \in \mathbb{R}^{N\times 7}$, where the last dimension corresponds to $(x, y, z, l, w, h, \theta)$ with object 3D centroids (x, y, z), headings θ in a BEV ego-relative coordinate frame, and the 3D box dimensions (w, l, h); multi-class confidence scores $\mathbf{C} \in [0, 1]^{N\times C}$, where C is the number of actor classes; trajectory forecasts $\mathbf{T} \in \mathbb{R}^{N\times T_f \times 3}$ describing objects' BEV pose $\{(x, y, \theta)_{t+s_f}, \ldots, (x, y, \theta)_{t+T_f s_f}\}$ over T_f future waypoints at a time interval s_f; and an object feature $\mathbf{Q} \in \mathbb{R}^{N\times (T_f+1)\times d}$ encoding both local and global features for every object at the present and future timestamps, where d

is the feature dimensionality. We use superscripts to denote the source of the proposals: detection proposals $\mathcal{P}^{\text{det}}$ from the 3D detector, memory proposals $\mathcal{P}^{\text{mem}}$ from the memory bank, merged proposals $\mathcal{P}^{\text{merge}}$ from the proposal merging module, and refined proposals $\mathcal{P}^{\text{ref}}$ from the output of the refinement transformer. For detection proposals $\mathcal{P}^{\text{det}}$, we generate $\mathbf{T}^{\text{det}}$ by assuming the object is static over time since detectors do not provide forecasts (and this is just an initialization before refinement). The object features $\mathbf{Q}^{\text{det}}$ are obtained by interpolating the feature map from before the detector header at the projected object centroids, repeating $(T_f + 1)$ times to get the features for future timestamps, and adding a learned embedding of $\mathbf{B}^{\text{det}}$ and $\mathbf{C}^{\text{det}}$. In the paragraphs below we describe how we obtain $\mathcal{P}^{\text{mem}}$, $\mathcal{P}^{\text{merge}}$ and $\mathcal{P}^{\text{ref}}$.

Memory Bank and Retrieval: The memory bank is a set of tuples $\left(t_m, \mathbf{E}_{t_m}, \mathcal{P}^{\text{ref}}_{t_m}\right)$ with timestamped past model outputs and ego pose, sorted by the timestamp t_m at which the outputs were generated. During inference at timestamp t, we retrieve memory entries $\mathcal{P}^{\text{ref}}_{t_m}$ at a set of past target timestamps $t_m \in \mathcal{T}_m$, where $\mathcal{T}_m = \{t - s_m, t - 2s_m, \ldots, t - T_m s_m\}$. T_m is the number of past target timestamps, and s_m is the time stride of the retrieved entries. To be precise, we retrieve the closest memory entry to each timestamp in $\mathcal{T}_m$ to be robust to small sensor delays.

Extracting Memory Proposals: For effective use of the memory at inference, we should align each retrieved entry $(t_m, \mathbf{E}_{t_m}, \mathcal{P}^{\text{ref}}_{t_m})$ in space and time with the current detection proposals at time t. We handle ego-motion by applying the relative transform $\mathbf{E}_{t_m \to t} = \mathbf{E}_t^{-1}\mathbf{E}_{t_m}$ to $\mathbf{B}^{\text{ref}}_{t_m}$ and $\mathbf{T}^{\text{ref}}_{t_m}$. To handle object motion, we linearly interpolate the stored trajectory forecast to the current timestamp t to obtain the proposal box $\mathbf{B}^{\text{mem}}$. To obtain the proposal forecast $\mathbf{T}^{\text{mem}}$, we also interpolate/extrapolate the stored trajectories as required to obtain waypoints at $\mathcal{T}_f = \{t + s_f, \ldots, t + T_f s_f\}$ from stored waypoints at $\{t_m + s_f, \ldots, t_m + T_f s_f\}$.

Finally, we extract latent features $\mathbf{Q}^{\text{mem}}$ at t and every future time step $t_f \in \mathcal{T}_f$: First, we compute sinusoidal positional embeddings [60] for the centroid coordinates $\mathbf{B}^{\text{mem}}_{x,y,z}$ and encode them with a lightweight MLP. Separately, we concatenate other features including $\mathbf{B}^{\text{mem}}_{l,w,h,\theta}$ (box dimensions and heading), confidence scores $\mathbf{C}^{\text{mem}}$, the memory age $t - t_m$, and a 2D vector pointing to where the proposal was in the current ego coordinate frame at the time t_m. Finally, we encode the concatenated features with another MLP and add the features from both MLPs together.

Proposal Merging: The memory and detection proposals can be redundant, particularly in areas with good sensor coverage. To merge proposals, we learn to rescore their multi-class confidence scores. Rescoring is essential as the confidence the model should put in a memory proposal not only depends on the confidence score at a past timestamp $\mathbf{C}^{\text{mem}}_{t_m}$, but also on the proposal age $t - t_m$, as the forecasting uncertainty grows with the time horizon and other factors. For example, the model should trust a fast-moving detection less than a stationary object, or it should trust an object observed 0.5 seconds ago more than one observed 5 seconds ago. Furthermore, the detection proposals come from the 3D detector, while the memory proposals are produced by MAD, and detectors have been found to be miscalibrated [24, 40, 42].

To make the scores comparable, we learn two small MLPs that separately map the features of the detection proposals $\mathbf{Q}^{\text{det}}$ and the memory proposals $\mathbf{Q}^{\text{mem}}$ to new multi-class scores $\mathbf{C}^{\text{merge}}$. As explained in Section 3.2, these rescoring MLPs are trained under a single detection loss applied to the merged proposals so that the model can decide which proposals to trust from both sources. Finally, we filter proposals with score thresholding, non-maximum suppression (NMS), and keep the top K *merged proposals* sorted by score (maximum over actor classes). Post-processing enables the refinement transformer to process a smaller number of queries.

Finally, we add learned time positional embeddings to the merged proposal features $\mathbf{Q}^{\text{merge}}$ to indicate the time of the trajectory forecast. At this point, we have $N^{\text{merge}} \stackrel{\text{def}}{=} N^{\text{ref}}$ merged proposals $\mathcal{P}^{\text{merge}}$ ready for refinement.

Refinement Transformer: We utilize a transformer decoder to refine the merged proposals $\mathcal{P}^{\text{merge}} = \mathcal{P}^{\text{ref}(0)}$ iteratively over I blocks into $\mathcal{P}^{\text{ref}(1)} \ldots \mathcal{P}^{\text{ref}(I)}$, where the final model outputs are $\mathcal{P}^{\text{ref}} = \mathcal{P}^{\text{ref}(I)}$. We propose a novel *memory cross-attention* mechanism to allow the queries — proposal features $\mathbf{Q}^{\text{ref}(i)}$ — to aggregate information from all the memory proposals $\mathbf{Q}^{\text{mem}}$, including those that proposal merging filtered out. We want to use this information in the refinement transformer because multiple overlapping memory proposals provide significant evidence about an object's presence and location. To achieve this, we perform cross attention from the object queries $\mathbf{Q}^{\text{ref}(i)}$ to the memory proposal features $\mathbf{Q}^{\text{mem}}$. For efficiency, we limit the cross attention to the nearest k keys to each object query (computing the nearest neighbors of $\mathbf{B}^{\text{ref}(i)}_{x,y,z}$ in $\mathbf{B}^{\text{mem}}_{x,y,z}$).

Similar to many works [4, 44], we perform factorized self-attention in each refinement block, which separates *time self-attention* and *object self-attention* for efficiency, where the former attends only to queries from the same object (sequence length $T_f + 1$) and the latter only attends to queries from the same time step (sequence length N). The updated queries $\mathbf{Q}^{\text{ref}(i+1)}$ are input to the next block.

Finally, we update the explicit proposal information as described in DeTra [4], by using a simple MLP to produce $\mathbf{B}^{\text{ref}(i+1)}$ and $\mathbf{C}^{\text{ref}(i+1)}$ and a gated recurrent unit (GRU) [9] to update the future trajectory waypoints $\mathbf{T}^{\text{ref}(i+1)}$.

Memory Bank Update: We post-process the refined proposals $\mathcal{P}^{\text{ref}}$ as we did to the merged proposals: score thresholding, NMS, and top K based on confidence score, adding the result to the memory bank, along with the corresponding timestamp t and ego pose $\mathbf{E}_t$. To limit the size of the memory bank when running on long sequences, we remove any memory entries older than $t - T_m s_m - \epsilon$ (the past time-horizon used in memory retrieval with a small buffer ϵ).

3.2. Training

We first train an off-the-shelf 3D detector following their original training strategy. This stage can be omitted if a pre-trained 3D detector is available. Then, we train all the parameters in MAD as a subsequent stage, with the 3D detector weights frozen. Pre-training and freezing the 3D detector is important to ensure the detection proposals do not change throughout MAD training. Note that we train a separate MAD for each 3D detector, as each detector has different features $\mathbf{Q}^{\text{det}}$ and detection distribution and calibration.

Before detailing our proposed MAD training, we discuss some possibilities and trade-offs when training temporal fusion models. Training on unordered examples has the advantage of satisfying the assumption of i.i.d examples (better learning dynamics) [21, 50]. However, it differs from evaluation, where the model is rolled out on long sequences and consumes its previous outputs. Training on long sequences of ordered data has the advantage of being closer to evaluation, but it has worse learning dynamics since consecutive examples are heavily correlated (there are few changes in the scene from one frame to the next). If, instead, gradients are accumulated over a long sequence and used to update the model parameters once per sequence, a sequence becomes one example (satisfying the i.i.d assumption), but the training duration is multiplied by the sequence length if the number of model updates is kept constant. Despite this large space of possibilities and the importance of such choices, prior works on learned temporal fusion neglect details and discussion of their training recipe [15, 17, 20, 30].

To tackle these challenges, we design a novel training schedule. We propose to train MAD on increasingly long chunks of ordered data, using single frames[1] at the beginning and entire sequences at the end of training. To train object memory on short chunks (or even single frames) of data while maintaining a reasonable amount of memory inputs, we propose to maintain a cache of memory banks across training and using it to build the memory proposals for each training example. Below, we detail this proposed schedule, our cache of memory banks, how we handle augmentations with memory, and our loss function.

Training Schedule: The datasets we use (WOD [57], AV2 [68]) organize their data into driving logs, each around 20s in duration with data captured at 10Hz, meaning each log has around 200 frames. Each log has a unique identifier (*logID*). For the first 25% of training, we sample single frames (that is, consecutive training examples are random frames from random logs). Throughout the rest of training, we sample sequential chunks of gradually increasing size: 48 frames for (25%, 50%] of training, 96 frames for (50%, 75%], and 144 frames for (75%, 100%]. We train with a single cosine decay learning rate schedule with no resets. The intuition behind this is that when the learning rate is high, and the model weights change the most, the model is exposed to more diverse data. Then, when the learning rate is lower, the model is tuned to be closer to the evaluation setting, where it consumes its previous outputs.

Exploiting a Cache of Memory Banks: If the schedule described above is followed naively during the individual frame and short chunk training, the model cannot consume its previous outputs and thus would not learn to use memory during this phase of training. To address this problem, we introduce a cache of previous memory banks. This cache is a mapping from the unique driving log identifier *logID* to a memory bank. At the start of training, we initialize the cache with empty memory banks for all *logID*s. On a given training iteration, we index the cache with the *logID* of the current training example to obtain the memory bank. If available, we retrieve the memory proposals from this memory bank as described in Sec. 3.1. We update the retrieved memory bank at the end of the training iteration with the model outputs, replacing any existing entry with the same timestamp. Note that during training we do not limit the size of the memory bank.

There are a few challenges to training with the object memory cache that we address:

- To train these models efficiently on large datasets, we use a distributed data training scheme, meaning we split examples in the minibatch across multiple GPUs. Each GPU has a unique index called a *rank*. Each rank maintains a separate cache to prevent the cache from filling up the RAM and avoid synchronization costs. To guarantee high cache hit rates, we ensure that training examples from a given *logID* are always put on the same rank during training.
- The cache is filled with MAD outputs, which are inaccurate at the beginning of training. We do not want erroneous model outputs to fill the cache; otherwise, the model may not learn to trust the memory proposals. To mitigate this, we only start filling the cache (and training with memory proposals) after 2.5% of training, after which performance is reasonable.
- To make the model robust to variable latency and the presence and absence of memory proposals, we randomize the target timestamps $\mathcal{T}_m$ that we retrieve memory elements for during training by randomly sampling the time stride

[1] We slightly abuse the term "frame" here, as some detectors use a window of multiple past frames as input.

Method	Overall L1		Overall L2		Vehicle L1		Vehicle L2		Pedestrian L1		Pedestrian L2		Cyclist L1		Cyclist L2	
	AP	APH	AP	APH	AP	APH	AP	APH	AP	APH	AP	APH	AP	APH	AP	APH
Centerpoint 1f [75]	76.1	73.5	70.0	67.6	75.7	75.2	67.9	67.4	77.6	71.6	70.1	64.4	74.9	73.8	72.1	71.0
+ MAD (Ours)	82.9	81.0	77.6	75.8	81.1	80.5	74.0	73.4	83.8	80.0	77.2	73.5	83.8	82.6	81.6	80.4
Centerpoint 2f [75]	77.5	75.8	71.7	70.1	76.4	75.9	68.7	68.2	79.2	75.6	71.9	68.5	76.8	75.9	74.4	73.5
+ MAD (Ours)	82.8	81.2	77.5	76.0	81.4	80.8	74.3	73.7	84.7	82.1	78.2	75.6	82.2	80.8	80.1	78.7
HEDNet 1f [81]	81.6	79.7	75.6	73.7	80.9	80.5	73.1	72.7	84.6	80.2	77.1	72.8	79.4	78.5	76.6	75.6
+ MAD (Ours)	85.2	83.3	80.2	78.3	83.6	82.9	76.6	76.0	87.0	83.4	81.0	77.4	85.1	83.7	83.0	81.6
HEDNet 4f [81]	83.6	82.3	78.1	76.8	82.4	81.9	75.1	74.6	86.3	83.6	79.4	76.8	82.2	81.4	79.9	79.1
+ MAD (Ours)	85.5	83.8	80.6	79.0	83.6	82.9	76.8	76.1	87.7	85.0	81.9	79.2	85.1	83.5	83.2	81.6
SAFDNet 1f [80]	81.7	79.7	75.5	73.6	80.5	80.0	72.5	72.1	84.7	80.2	77.1	72.9	79.8	78.8	76.9	75.9
+ MAD (Ours)	85.3	83.5	80.3	78.4	83.4	82.8	76.5	75.9	86.8	82.9	80.7	76.8	**85.8**	**84.7**	**83.7**	**82.6**
SAFDNet 4f [80]	83.9	82.6	78.4	77.1	82.8	82.3	75.4	74.9	86.8	84.2	80.1	77.5	82.0	81.1	79.6	78.8
+ MAD (Ours)	**85.8**	**84.2**	**81.0**	**79.4**	**84.2**	**83.6**	**77.4**	**76.8**	**87.9**	**85.4**	**82.2**	**79.7**	85.2	83.7	83.3	81.7

Table 1. Comparing the performance of various off-the-shelf LiDAR object detectors with and without MAD on the WOD validation set. Base detector results are reproduced using official code. MAD consistently boosts the performance of all detectors across all metrics.

Method	Vehicle AP IoU 10 In Camera Field of View			
	Overall	[0, 40)m	[40, 80)m	[80, 120)m
FCOS3D [64]	37.6	73.9	34.3	4.65
+MAD (Ours)	**43.6**	**82.6**	**40.0**	**8.11**
BEVMap [6]	51.5	86.5	54.0	13.9
+MAD (Ours)	**53.4**	**88.0**	**55.1**	**17.2**

Table 2. Adding MAD to camera-based 3D detectors on AV2.

s_m, and the number of target timestamps T_m.

Handling Augmentations with Memory: Prior works [26, 75, 80, 81] find that data augmentations (e.g., translation, rotation, flipping, and re-scaling) are important for detection performance. We apply augmentations to the boxes $\mathbf{B}^{\text{mem}}$ and trajectories $\mathbf{T}^{\text{mem}}$ in memory proposals after the memory alignment step. We apply the inverse of the augmentations to $\mathbf{B}^{\text{ref}}$ and $\mathbf{T}^{\text{ref}}$ before storing them in the memory bank.

Loss function: We optimize a multi-task objective $L = L_{\text{rescore}}(\mathbf{C}^{\text{merge}}) + \sum_{i=1}^{I} L_{\text{det}}(\mathbf{B}^{\text{ref}(i)}, \mathbf{C}^{\text{ref}(i)}) + L_{\text{for}}(\mathbf{T}^{\text{ref}(i)})$, which is a combination of a rescoring loss L_{score}, a detection refinement loss L_{det}, and a forecasting refinement loss L_{for}, where the detection and forecasting losses are computed at every refinement block. Following [4], L_{det} includes a binary focal loss for classification, an L1 loss for regression and an IoU loss. To calculate the targets for these losses, we first match the detections to the ground truth bounding boxes through bipartite matching as proposed in DETR [3]. The rescoring loss is similar, except it consists only of the focal loss as we are only training the multi-class scores $\mathbf{C}^{\text{merge}}$ output by the rescoring module. The trajectory refinement loss is an L1 loss against the ground-truth trajectory, supervised only for true-positive detections (with IoU with a ground truth box higher than 0.5). See our supplementary for more details.

Method	AP L1	APH L1	AP L2	APH L2
CenterFormer [82]	82.3	80.9	77.6	76.3
BEVFusion [36]	82.7	81.4	77.7	76.3
MSF [15]	83.1	81.7	78.3	77.0
FSD++ [13]	83.5	82.1	78.4	77.1
LoGoNet [29]	83.1	81.8	78.4	77.1
Octopus_Noah	83.1	81.7	78.7	77.3
SEED-L [38]	83.5	82.2	78.7	77.3
LION [37]	83.7	82.4	78.7	77.4
VeuronNet3D	83.7	82.2	79.1	77.7
HIAC	84.0	82.6	79.2	77.8
InceptioLidar	83.8	82.5	79.2	77.8
VADet	84.1	82.8	79.4	78.2
MT3D	85.0	83.7	80.1	78.7
LIVOX Detection	84.8	83.5	80.2	79.0
MAD (Ours)	**86.0**	**84.3**	**81.8**	**80.2**

Table 3. Results on the WOD test set, as reported on the leaderboard[2]. We exclude entries that state they use ensembles, test-time augmentations, or are offline (use future sensor data). "Ours" is using SAFDNet 4f as the 3D detector. APH L2 is the ranking metric.

4 Experiments

This section provides a comprehensive quantitative analysis of MAD from three perspectives. First, we add MAD to existing 3D detectors, showing significant improvements. We use both LiDAR-based and camera-based detectors on WOD [57] and AV2 [68], respectively. Second, we compare the best version of MAD to the state-of-the-art methods on WOD, setting a new record on the WOD leaderboard among online methods without ensembles or test-time augmentation and outperforming prior learned temporal fusion methods by a large margin. Finally, we conduct thorough

	Method	AP L1	APH L1	AP L2	APH L2
Validation	LEF [17]	79.6	79.2	71.4	70.9
	MoDAR [30]	-	-	-	72.5
	MPPNet [8]	81.6	81.1	76.0	74.8
	MSF [15]	82.2	80.7	76.8	75.5
	PTT [20]	82.7	80.7	77.7	75.7
	MAD (Ours)	**85.8**	**84.2**	**81.0**	**79.4**
Testing	3D-MAN [74]	49.6	48.1	44.8	43.4
	MPPNet [8]	81.8	80.6	76.9	75.7
	MSF [15]	83.1	81.7	78.3	77.0
	MAD (Ours)	**86.0**	**84.3**	**81.8**	**80.2**

Table 4. Comparison of our method against various methods for learned temporal fusion on WOD. "Ours" is using SAFDNet 4f.

ablation studies to understand the architectural choices that make MAD effective and the impact of different training procedures. Refer to our supplementary for more implementation details, experimental results, and ablations.

Implementation Details: The refinement transformer uses $I = 3$ refinement blocks, and the dimension of all embeddings is $d = 128$. We forecast $T_f = 10$ future timestamps at stride of $s_f = 0.5$s, yielding a 5s prediction horizon. Unless otherwise stated, we use target timestamps of $\mathcal{T}_m = \{-0.3s, -0.6s, \ldots, -2.4s\}$ (i.e., $s_m = 0.3$s, $T_m = 8$) for reading from the memory bank at inference. In the memory cross attention we use the nearest $k = 4$ neighbors. Following prior works [75, 80, 81], for any detection post-processing, we use a 0.1 confidence threshold; per-class NMS IoU thresholds of $\{0.75, 0.6, 0.55\}$ for vehicles, pedestrians, and cyclists, respectively; and a top $K = 500$. MAD has 3.8M parameters, while the base detectors have anywhere from 8M (Centerpoint [75]) to 53M (BEVMap [6]) parameters. For each base detector, we train MAD for 60k update steps (roughly equivalent to 6 epochs on WOD and AV2), with batch size 16. We use a cosine learning rate decay with a max learning rate of 8×10^{-4}, and a linear warm-up for the first 1000 steps, beginning with a learning rate of 8×10^{-5}. During training, we use a variable set of memory target timestamps $T_m \sim \texttt{uniform}(\{6, 7, 8, 9, 10\})$ with a variable stride $s_m \sim \texttt{uniform}(\{0.2s, 0.3s, 0.4s\})$.

Metrics: We report the detection metrics from the official WOD leaderboard [57], which include average precision (AP) and AP weighted by heading error (APH) for vehicles (Veh.), pedestrians (Ped.), and cyclists (Cyc.). These metrics use intersection-over-union (IoU) thresholds of 0.7, 0.5, and 0.5, respectively. The metrics are broken down into two levels of difficulty: Level 1 (L1) includes only labels that have > 5 LiDAR points and are not marked as "hard", and Level (L2) includes all boxes that have > 0 LiDAR points (a superset of L1). For camera experiments on AV2, we report the mean average precision (AP) for vehicles in the camera field of view at an IoU threshold of 0.1. We report the macro-average over all classes if the actor class is not specified.

Augmenting off-the-shelf 3D Detectors with MAD: Tab. 1 and Tab. 2 show the performance of MAD applied to off-the-shelf 3D detectors on WOD and AV2, respectively.To show the generality of our approach, we experiment with multiple base detectors trained on different datasets and sensor modalities. We enhanced three LiDAR-based methods with MAD on WOD: CenterPoint [75] with both 1 LiDAR frame (1f) and 2 LiDAR frames (2f) as input, HEDNet [81] (1f and 4f), and SAFDNet [80] (1f and 4f). We follow their official protocols to train and evaluate all models from scratch (due to the Waymo Dataset License Agreement, we cannot simply re-use pre-trained models). We also enhance two camera-based methods on AV2, FCOS3D [64] and BEVMap [6], which takes the most recent image from the front camera as input. We use the official implementation for both FCOS3D [64] and BEVMap [6]. Training details are in the supplementary.

Our model brings significant improvements to all detectors on both datasets. These gains are largest for single-frame detectors, where the memory provides the most additional information. The fact that the MAD-augmented single-frame detectors are better than the multi-frame detectors clearly shows the effectiveness of our method relative to the common point aggregation approach. Please visit the supplementary materials for qualitative comparisons.

Comparison against SOTA: By augmenting SAFDNet 4f with MAD, we show in Tab. 3 that we achieve the best performance on the WOD leaderboard[2], among all online methods that do not use ensembles or test-time augmentation. Table 4 compares MAD to prior learned temporal fusion methods on the WOD validation and test set, where we achieve substantial gains. Please refer to our supplementary for full Tabs. 3 and 4 with metrics for all actor classes.

Effect of memory proposals and memory attention: We ablate the different components of our memory pipeline in Tab. 5. Comparing rows 1, 2, and 5 shows that both the proposed memory attention and memory proposals have a positive effect. This is intuitive as the memory proposals let MAD recover from false negative detection proposals, which is complementary to memory cross-attention that allows MAD to use all memory information for refinement (bypassing the filtering in proposal-merging).

Effect of forecasting: Comparing rows 3 and 5 in Tab. 5, we find that using trajectory forecasting to align memory proposals to the current time is important, particularly for fast-moving objects. Without forecasting, the memory proposals from previous frames will be far from the current po-

[2] https://waymo.com/open/challenges/2020/3d-detection/ as of submission (14/11/2024)

	Mem. Prop.	Mem. Attn.	Forecast.	Rescore.	Veh. AP	Ped. AP	Cyc. AP	Veh. AP [20, 30) m/s	Cyc. AP [5, 10) m/s
0	✗	✗	✗	✗	75.4	80.1	79.6	38.0	72.4
1	✗	✓	✓	✓	76.5	81.3	82.0	40.4	82.1
2	✓	✗	✓	✓	75.8	81.7	81.6	37.7	75.9
3	✓	✓	✗	✓	76.9	81.8	81.9	34.6	78.7
4	✓	✓	✓	✗	72.7	82.0	81.2	11.9	73.9
5	✓	✓	✓	✓	**77.0**	**82.3**	**83.3**	**45.2**	**86.2**

Table 5. Component ablation of MAD on the WOD validation set. All metrics are L2. Row 0 is the base 3D detector, SAFDNet 4f [80]. All ablations in this table (including the final method with all components) use a reduced training duration of 45k iterations to reduce costs.

Evaluated Proposals	AP L1	APH L1	AP L2	APH L2
Detection $(\mathcal{P}^{\text{det}})$	83.9	82.6	78.4	77.1
Combined $(\mathcal{P}^{\text{mem}} \cup \mathcal{P}^{\text{det}})$	18.1	17.8	16.8	16.4
Merged $(\mathcal{P}^{\text{merge}})$	84.1	83.0	78.8	77.7
After Block 0 $(\mathcal{P}^{\text{ref}(1)})$	84.4	83.2	79.3	78.1
After Block 1 $(\mathcal{P}^{\text{ref}(2)})$	85.8	84.2	80.9	79.3
After Block 2 $(\mathcal{P}^{\text{ref}})$	**85.8**	**84.2**	**81.0**	**79.4**

Table 6. Evaluating various intermediate proposals from MAD. The base detector is SAFDNet 4f.

	Chunk Length	Cache	AP L1	APH L1	AP L2	APH L2
1a	144	✗	85.4	83.9	80.5	79.0
1b	144	✓	85.2	83.7	80.3	78.9
2a	48	✗	85.1	83.5	80.2	78.6
2b	48	✓	85.3	83.8	80.3	78.9
3a	1	✗	83.9	82.4	78.4	77.0
3b	1	✓	85.0	83.3	80.1	78.5
4a	1→48→96→144	✗	84.9	83.3	79.9	78.3
4b	1→48→96→144	✓	**85.8**	**84.2**	**81.0**	**79.4**

Table 7. Ablating chunk length and the memory cache on WOD, using SAFDNet 4f.

sition of those objects, making it challenging for the model to leverage the memory effectively.

Effect of learned proposal merging: Comparing rows 0, 4, and 5 of Tab. 5 we find the proposed learned rescoring of the merged detection and memory proposals is crucial for good performance. Without it, MAD cannot enhance the base detector (row 0) because the proposal scores from the 3D detector and memory are miss-calibrated before being post-processed in the proposal merging step (i.e., NMS). We illustrate this in Tab. 6, where we evaluate intermediate proposals of MAD: (1) the detection proposals $\mathcal{P}^{\text{det}}$, (2) naively taking the union of the detection proposals $\mathcal{P}^{\text{det}}$ and memory proposals $\mathcal{P}^{\text{mem}}$ and post-processing them, (3) the merged proposals $\mathcal{P}^{\text{merge}}$ (which have been rescored), and (4) after each block of the refinement transformer $\mathcal{P}^{\text{ref}(1)}, \ldots, \mathcal{P}^{\text{ref}(I)}$. Naively concatenating the combined proposals is much worse than the base detector because of the miss-calibrated scores. After proposal merging, $\mathcal{P}^{\text{merge}}$ already improves over the base detector. Each refinement block brings further gains, illustrating the strength of our proposed refinement transformer.

Training Procedure Study: Table 7 provides evidence supporting the effectiveness of our proposed training schedule. We first train MAD with three different chunk sizes (i.e., sequences with $\{144, 48, 1\}$ frames), each with and without the memory bank cache. Training with long chunks (144 frames, Tab. 7.1a) provides good performance because there is a low gap between training and evaluation. The cache provides no gains in this setting (Tab. 7.1b) because the model already has memory proposals in most frames. Training with shorter chunks (Tab. 7.2a,3a) performs worse because there is a more significant gap between training and evaluation. Including the cache helps significantly by closing the gap to evaluation but does not fully reach the long chunk performance (Tab. 7.2b,3b). As hypothesized in Sec. 3.2, there is room for improvement by training with our proposed schedule and memory bank cache (Tab. 7.4b). This strategy allows MAD to learn generalized patterns over a diverse set of examples quickly by training on short chunks (more i.i.d. data) at the beginning when the learning rate is higher while refining its understanding on long chunks (closer to the deployment setting) towards the end when the learning rate is lower. Table 7.4a shows the importance of the cache when using this training schedule; otherwise, training with small chunks is ineffective as the model would not learn to use the memory.

5 Conclusion

In this paper, we propose MAD — a simple, effective, and sensor-modality-agnostic add-on for enhancing any existing 3D object detector with long-term memory. To achieve this, we design a transformer-based model that uses joint detection and trajectory forecasting to populate a memory bank with spatial-temporal object trajectories. Our model can effectively fuse memory proposals with detection proposals by reading previous memory entries and aligning them with the current time and ego pose. We also propose a novel training strategy that increases data diversity while keeping the training-to-inference gap low. Our approach is very general — bringing impressive improvements to a variety of LiDAR-based and camera-based detectors, and very effective — achieving SOTA performance on Waymo Open Dataset when paired to the base detector SAFDNet 4f [80].

References

[1] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020. 1

[2] Yigit Baran Can, Alexander Liniger, Ozan Unal, Danda Paudel, and Luc Van Gool. Understanding bird's-eye view of road semantics using an onboard camera. *IEEE Robotics and Automation Letters*, 7(2):3302–3309, 2022. 2

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020. 6, 1

[4] Sergio Casas, Ben Agro, Jiageng Mao, Thomas Gilles, Alexander Cui, Thomas Li, and Raquel Urtasun. Detra: A unified model for object detection and trajectory forecasting. 2024. 2, 4, 6, 5

[5] Yuning Chai, Benjamin Sapp, Mayank Bansal, and Dragomir Anguelov. Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction. *arXiv preprint arXiv:1910.05449*, 2019. 2

[6] Mincheol Chang, Seokha Moon, Reza Mahjourian, and Jinkyu Kim. Bevmap: Map-aware bev modeling for 3d perception. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 7419–7428, 2024. 2, 6, 7, 1

[7] Xiaozhi Chen, Kaustav Kundu, Yukun Zhu, Andrew G Berneshawi, Huimin Ma, Sanja Fidler, and Raquel Urtasun. 3d object proposals for accurate object class detection. *Advances in neural information processing systems*, 28, 2015. 2

[8] 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 *European Conference on Computer Vision*, pages 680–697. Springer, 2022. 1, 2, 3, 7

[9] Kyunghyun Cho. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014. 4, 1

[10] Alexander Cui, Sergio Casas, Kelvin Wong, Simon Suo, and Raquel Urtasun. Gorela: Go relative for viewpoint-invariant motion forecasting. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7801–7807. IEEE, 2023. 2, 5

[11] Henggang Cui, Vladan Radosavljevic, Fang-Chieh Chou, Tsung-Han Lin, Thi Nguyen, Tzu-Kuo Huang, Jeff Schneider, and Nemanja Djuric. Multimodal trajectory predictions for autonomous driving using deep convolutional networks. In *2019 international conference on robotics and automation (icra)*, pages 2090–2096. IEEE, 2019. 2

[12] Martin Engelcke, Dushyant Rao, Dominic Zeng Wang, Chi Hay Tong, and Ingmar Posner. Vote3deep: Fast object detection in 3d point clouds using efficient convolutional neural networks. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1355–1361. IEEE, 2017. 2

[13] Lue Fan, Yuxue Yang, Feng Wang, Naiyan Wang, and Zhaoxiang Zhang. Super sparse 3d object detection. *IEEE transactions on pattern analysis and machine intelligence*, 45(10):12490–12505, 2023. 6, 3

[14] Davi Frossard, Shun Da Suo, Sergio Casas, James Tu, and Raquel Urtasun. Strobe: Streaming object detection from lidar packets. In *Conference on Robot Learning*, pages 1174–1183. PMLR, 2021. 1, 2

[15] Chenhang He, Ruihuang Li, Yabin Zhang, Shuai Li, and Lei Zhang. Msf: Motion-guided sequential fusion for efficient 3d object detection from point cloud sequences. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5196–5205, 2023. 1, 2, 3, 5, 6, 7

[16] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 2

[17] Tong He, Pei Sun, Zhaoqi Leng, Chenxi Liu, Dragomir Anguelov, and Mingxing Tan. Lef: Late-to-early temporal fusion for lidar 3d object detection. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1637–1644. IEEE, 2023. 1, 2, 5, 7, 3

[18] Jinghua Hou, Zhe Liu, Zhikang Zou, Xiaoqing Ye, Xiang Bai, et al. Query-based temporal fusion with explicit motion for 3d object detection. *Advances in Neural Information Processing Systems*, 36, 2024. 1, 3

[19] Anthony Hu, Zak Murez, Nikhil Mohan, Sofía Dudas, Jeffrey Hawke, Vijay Badrinarayanan, Roberto Cipolla, and Alex Kendall. Fiery: Future instance prediction in bird's-eye view from surround monocular cameras. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15273–15282, 2021. 2

[20] Kuan-Chih Huang, Weijie Lyu, Ming-Hsuan Yang, and Yi-Hsuan Tsai. Ptt: Point-trajectory transformer for efficient temporal 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14938–14947, 2024. 1, 2, 3, 5, 7

[21] Jack Kiefer and Jacob Wolfowitz. Stochastic estimation of the maximum of a regression function. *The Annals of Mathematical Statistics*, pages 462–466, 1952. 5

[22] Junho Koh, Junhyung Lee, Youngwoo Lee, Jaekyum Kim, and Jun Won Choi. Mgtanet: Encoding sequential lidar points using long short-term motion-guided temporal attention for 3d object detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 1179–1187, 2023. 2

[23] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven L Waslander. Joint 3d proposal generation and object detection from view aggregation. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–8. IEEE, 2018. 2

[24] Fabian Kuppers, Jan Kronenberger, Amirhossein Shantia, and Anselm Haselhoff. Multivariate confidence calibration for object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 326–327, 2020. 4

[25] 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/CVF conference on computer vision and pattern recognition*, pages 12697–12705, 2019. 2

[26] Zhaoqi Leng, Guowang Li, Chenxi Liu, Ekin Dogus Cubuk, Pei Sun, Tong He, Dragomir Anguelov, and Mingxing Tan. Lidar augment: Searching for scalable 3d lidar data augmentations. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7039–7045. IEEE, 2023. 6

[27] Bo Li. 3d fully convolutional network for vehicle detection in point cloud. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1513–1518. IEEE, 2017. 2

[28] Hongyang Li, Hao Zhang, Zhaoyang Zeng, Shilong Liu, Feng Li, Tianhe Ren, and Lei Zhang. Dfa3d: 3d deformable attention for 2d-to-3d feature lifting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6684–6693, 2023. 2

[29] Xin Li, Tao Ma, Yuenan Hou, Botian Shi, Yuchen Yang, Youquan Liu, Xingjiao Wu, Qin Chen, Yikang Li, Yu Qiao, et al. Logonet: Towards accurate 3d object detection with local-to-global cross-modal fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17524–17534, 2023. 6

[30] Yingwei Li, Charles R Qi, Yin Zhou, Chenxi Liu, and Dragomir Anguelov. Modar: Using motion forecasting for 3d object detection in point cloud sequences. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9329–9339, 2023. 1, 2, 5, 7, 3

[31] Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Yu Qiao, and Jifeng Dai. Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. In *European conference on computer vision*, pages 1–18. Springer, 2022. 1, 2

[32] Ming Liang, Bin Yang, Wenyuan Zeng, Yun Chen, Rui Hu, Sergio Casas, and Raquel Urtasun. Pnpnet: End-to-end perception and prediction with tracking in the loop. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11553–11562, 2020. 1, 2

[33] Xuewu Lin, Tianwei Lin, Zixiang Pei, Lichao Huang, and Zhizhong Su. Sparse4d: Multi-view 3d object detection with sparse spatial-temporal fusion. *arXiv preprint arXiv:2211.10581*, 2022. 2

[34] Yingfei Liu, Tiancai Wang, Xiangyu Zhang, and Jian Sun. Petr: Position embedding transformation for multi-view 3d object detection. In *European Conference on Computer Vision*, pages 531–548. Springer, 2022.

[35] Yingfei Liu, Junjie Yan, Fan Jia, Shuailin Li, Aqi Gao, Tiancai Wang, and Xiangyu Zhang. Petrv2: A unified framework for 3d perception from multi-camera images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3262–3272, 2023. 2

[36] Zhijian Liu, Haotian Tang, Alexander Amini, Xinyu Yang, Huizi Mao, Daniela L Rus, and Song Han. Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation. In *2023 IEEE international conference on robotics and automation (ICRA)*, pages 2774–2781. IEEE, 2023. 6, 3

[37] Zhe Liu, Jinghua Hou, Xinyu Wang, Xiaoqing Ye, Jingdong Wang, Hengshuang Zhao, and Xiang Bai. Lion: Linear group rnn for 3d object detection in point clouds. *arXiv preprint arXiv:2407.18232*, 2024. 6, 3

[38] Zhe Liu, Jinghua Hou, Xiaoqing Ye, Tong Wang, Jingdong Wang, and Xiang Bai. Seed: A simple and effective 3d detr in point clouds. *arXiv preprint arXiv:2407.10749*, 2024. 6, 3

[39] 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. 1

[40] Matthias Minderer, Josip Djolonga, Rob Romijnders, Frances Hubis, Xiaohua Zhai, Neil Houlsby, Dustin Tran, and Mario Lucic. Revisiting the calibration of modern neural networks. *Advances in Neural Information Processing Systems*, 34:15682–15694, 2021. 4

[41] Arsalan Mousavian, Dragomir Anguelov, John Flynn, and Jana Kosecka. 3d bounding box estimation using deep learning and geometry. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 7074–7082, 2017. 2

[42] Lukas Neumann, Andrew Zisserman, and Andrea Vedaldi. Relaxed softmax: Efficient confidence auto-calibration for safe pedestrian detection. 2018. 4

[43] 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. 2

[44] Jiquan Ngiam, Benjamin Caine, Vijay Vasudevan, Zhengdong Zhang, Hao-Tien Lewis Chiang, Jeffrey Ling, Rebecca Roelofs, Alex Bewley, Chenxi Liu, Ashish Venugopal, et al. Scene transformer: A unified architecture for predicting multiple agent trajectories. *arXiv preprint arXiv:2106.08417*, 2021. 4, 2

[45] Tung Phan-Minh, Elena Corina Grigore, Freddy A Boulton, Oscar Beijbom, and Eric M Wolff. Covernet: Multimodal behavior prediction using trajectory sets. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14074–14083, 2020. 2

[46] AJ Piergiovanni, Vincent Casser, Michael S Ryoo, and Anelia Angelova. 4d-net for learned multi-modal alignment. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15435–15445, 2021. 1, 2

[47] Rui Qian, Divyansh Garg, Yan Wang, Yurong You, Serge Belongie, Bharath Hariharan, Mark Campbell, Kilian Q Weinberger, and Wei-Lun Chao. End-to-end pseudo-lidar for image-based 3d object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5881–5890, 2020. 2

[48] Cody Reading, Ali Harakeh, Julia Chae, and Steven L Waslander. Categorical depth distribution network for monocular 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8555–8564, 2021. 2

[49] Shaoqing Ren. Faster r-cnn: Towards real-time object detection with region proposal networks. *arXiv preprint arXiv:1506.01497*, 2015. 2

[50] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951. 5

[51] Thomas Roddick, Alex Kendall, and Roberto Cipolla. Orthographic feature transform for monocular 3d object detection. *arXiv preprint arXiv:1811.08188*, 2018. 2

[52] Avishkar Saha, Oscar Mendez, Chris Russell, and Richard Bowden. Translating images into maps. In *2022 International conference on robotics and automation (ICRA)*, pages 9200–9206. IEEE, 2022. 2

[53] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 770–779, 2019. 2

[54] Weijing Shi and Raj Rajkumar. Point-gnn: Graph neural network for 3d object detection in a point cloud. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1711–1719, 2020. 2

[55] Martin Simony, Stefan Milzy, Karl Amendey, and Horst-Michael Gross. Complex-yolo: An euler-region-proposal for real-time 3d object detection on point clouds. In *Proceedings of the European conference on computer vision (ECCV) workshops*, pages 0–0, 2018. 2

[56] Shuran Song and Jianxiong Xiao. Deep sliding shapes for amodal 3d object detection in rgb-d images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 808–816, 2016. 2

[57] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *CVPR*, 2020. 2, 3, 5, 6, 7

[58] 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. In *European Conference on Computer Vision*, pages 426–442. Springer, 2022. 1

[59] Balakrishnan Varadarajan, Ahmed Hefny, Avikalp Srivastava, Khaled S Refaat, Nigamaa Nayakanti, Andre Cornman, Kan Chen, Bertrand Douillard, Chi Pang Lam, Dragomir Anguelov, et al. Multipath++: Efficient information fusion and trajectory aggregation for behavior prediction. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 7814–7821. IEEE, 2022. 2

[60] A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017. 4

[61] Dominic Zeng Wang and Ingmar Posner. Voting for voting in online point cloud object detection. In *Robotics: science and systems*, pages 10–15. Rome, Italy, 2015. 2

[62] Haiyang Wang, Chen Shi, Shaoshuai Shi, Meng Lei, Sen Wang, Di He, Bernt Schiele, and Liwei Wang. Dsvt: Dynamic sparse voxel transformer with rotated sets. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13520–13529, 2023. 1

[63] Shihao Wang, Yingfei Liu, Tiancai Wang, Ying Li, and Xiangyu Zhang. Exploring object-centric temporal modeling for efficient multi-view 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3621–3631, 2023. 3

[64] Tai Wang, Xinge Zhu, Jiangmiao Pang, and Dahua Lin. Fcos3d: Fully convolutional one-stage monocular 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 913–922, 2021. 2, 6, 7, 1

[65] Yan Wang, Wei-Lun Chao, Divyansh Garg, Bharath Hariharan, Mark Campbell, and Kilian Q Weinberger. Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8445–8453, 2019. 2

[66] Yue Wang, Alireza Fathi, Abhijit Kundu, David A Ross, Caroline Pantofaru, Tom Funkhouser, and Justin Solomon. Pillar-based object detection for autonomous driving. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXII 16*, pages 18–34. Springer, 2020. 2

[67] Yue Wang, Vitor Campagnolo Guizilini, Tianyuan Zhang, Yilun Wang, Hang Zhao, and Justin Solomon. Detr3d: 3d object detection from multi-view images via 3d-to-2d queries. In *Conference on Robot Learning*, pages 180–191. PMLR, 2022. 2

[68] Benjamin Wilson, William Qi, Tanmay Agarwal, John Lambert, Jagjeet Singh, Siddhesh Khandelwal, Bowen Pan, Ratnesh Kumar, Andrew Hartnett, Jhony Kaesemodel Pontes, et al. Argoverse 2: Next generation datasets for self-driving perception and forecasting. *arXiv preprint arXiv:2301.00493*, 2023. 2, 3, 5, 6

[69] Xiaoyang Wu, Li Jiang, Peng-Shuai Wang, Zhijian Liu, Xihui Liu, Yu Qiao, Wanli Ouyang, Tong He, and Hengshuang Zhao. Point transformer v3: Simpler faster stronger. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4840–4851, 2024. 2

[70] Bin Xu and Zhenzhong Chen. Multi-level fusion based 3d object detection from monocular images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2345–2353, 2018. 2

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

[72] Bin Yang, Wenjie Luo, and Raquel Urtasun. Pixor: Real-time 3d object detection from point clouds. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 7652–7660, 2018. 2

[73] 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*, pages 11040–11048, 2020. 2

[74] 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*, pages 1863–1872, 2021. 1, 3, 7

[75] 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*, pages 11784–11793, 2021. 1, 2, 6, 7

[76] Yurong You, Yan Wang, Wei-Lun Chao, Divyansh Garg, Geoff Pleiss, Bharath Hariharan, Mark Campbell, and Kilian Q Weinberger. Pseudo-lidar++: Accurate depth for 3d object detection in autonomous driving. *arXiv preprint arXiv:1906.06310*, 2019. 2

[77] Yurong You, Katie Z Luo, Xiangyu Chen, Junan Chen, Wei-Lun Chao, Wen Sun, Bharath Hariharan, Mark Campbell, and Kilian Q Weinberger. Hindsight is 20/20: Leveraging past traversals to aid 3d perception. *arXiv preprint arXiv:2203.11405*, 2022. 2

[78] Yihan Zeng, Da Zhang, Chunwei Wang, Zhenwei Miao, Ting Liu, Xin Zhan, Dayang Hao, and Chao Ma. Lift: Learning 4d lidar image fusion transformer for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17172–17181, 2022. 1, 2

[79] Ce Zhang, Chengjie Zhang, Yiluan Guo, Lingji Chen, and Michael Happold. Motiontrack: end-to-end transformer-based multi-object tracking with lidar-camera fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 151–160, 2023. 2

[80] Gang Zhang, Junnan Chen, Guohuan Gao, Jianmin Li, Si Liu, and Xiaolin Hu. Safdnet: A simple and effective network for fully sparse 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14477–14486, 2024. 2, 6, 7, 8, 1

[81] Gang Zhang, Chen Junnan, Guohuan Gao, Jianmin Li, and Xiaolin Hu. Hednet: A hierarchical encoder-decoder network for 3d object detection in point clouds. *Advances in Neural Information Processing Systems*, 36, 2024. 2, 6, 7, 1

[82] Zixiang Zhou, Xiangchen Zhao, Yu Wang, Panqu Wang, and Hassan Foroosh. Centerformer: Center-based transformer for 3d object detection. In *European Conference on Computer Vision*, pages 496–513. Springer, 2022. 2, 3, 6