Offboard 3D Object Detection from Point Cloud Sequences

Charles R. Qi Yin Zhou Mahyar Najibi Pei Sun Khoa Vo Boyang Deng Dragomir Anguelov
Waymo LLC

Abstract

While current 3D object recognition research mostly focuses on the real-time, onboard scenario, there are many offboard use cases of perception that are largely under-explored, such as using machines to automatically generate high-quality 3D labels. Existing 3D object detectors fail to satisfy the high-quality requirement for offboard uses due to the limited input and speed constraints. In this paper, we propose a novel offboard 3D object detection pipeline using point cloud sequence data. Observing that different frames capture complementary views of objects, we design the offboard detector to make use of the temporal points through both multi-frame object detection and novel object-centric refinement models. Evaluated on the Waymo Open Dataset, our pipeline named 3D Auto Labeling shows significant gains compared to the state-of-the-art onboard detectors and our offboard baselines. Its performance is even on par with human labels verified through a human label study. Further experiments demonstrate the application of auto labels for semi-supervised learning and provide extensive analysis to validate various design choices.

1. Introduction

Recent years have seen a rapid progress of 3D object recognition with advances in 3D deep learning and strong application demands. However, most 3D perception research has been focusing on real-time, onboard use cases and only considers sensor input from the current frame or a few history frames. Those models are sub-optimal for many *offboard* use cases where the best perception quality is needed. Among them, one important direction is to have machines "auto label" the data to save the cost of human labeling. High quality perception can also be used for simulation or to build datasets to supervise or evaluate downstream modules such as behavior prediction.

In this paper, we propose a novel pipeline for offboard 3D object detection with a modular design and a series of tailored deep network models. The offboard pipeline makes use of the whole sensor sequence input (such video data is common in applications of autonomous driving and augmented reality). With no constraints on the model causality and little constraint on model inference speed, we are able to greatly expand the design space of 3D object detectors and achieve significantly higher performance.

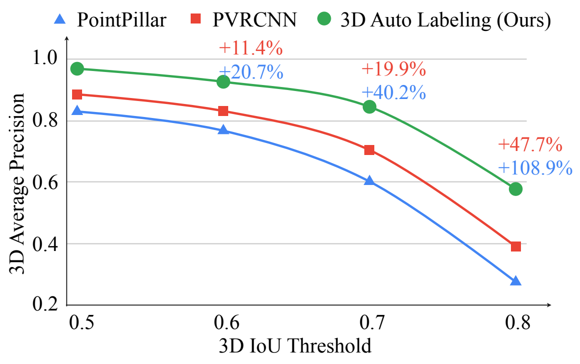

Figure 1. **Our offboard 3D Auto Labeling achieved significant gains over two representative onboard 3D detectors** (the efficient PointPillar [20] and the top-performing PVRCNN [45]). The relative gains (the percentage numbers) are higher under more strict standard (higher IoU thresholds). The metric is 3D AP (L1) for vehicles on the Waymo Open Dataset [52] *val* set.

We design our offboard 3D detector based on a key observation: different viewpoints of an object, within a point cloud sequence, contain complementary information about its geometry (Fig. 2). An immediate baseline design is to extend the current detectors to use multi-frame inputs. However, as multi-frame detectors are effective they are still limited in the amount of context they can use and are not naively scalable to more frames – gains from adding more frames diminish quickly (Table 5).

In order to fully utilize temporal point clouds (*e.g.* 10 or more seconds), we step away from the common frame-based input structure where the entire frames of point clouds are merged. Instead, we turn to an *object-centric* design. We first leverage a top-performing multi-frame detector to give us initial object localization. Then, we link objects detected at different frames through multi-object tracking. Based on the tracked boxes and the raw point cloud sequences, we can extract the entire track data of an object, including all of its sensor data (point clouds) and detector boxes, which is 4D: 3D spatial plus 1D temporal. We then propose novel deep network models to process such 4D object track data and output temporally consistent and high-quality boxes of the object. As they are similar to how a human labels an object and because of their high-quality output, we call those models processing the 4D track data as "object-centric auto labeling models" and the entire pipeline "3D Auto Labeling" (Fig. 3).

We evaluate our proposed models on the Waymo Open Dataset (WOD) [52] which is a large-scale autonomous driving benchmark containing 1,000+ Lidar scan sequences with 3D annotations for every frame. Our 3D Auto Labeling pipeline dramatically lifts the perception quality compared to existing 3D detectors designed for the real-time, onboard use cases (Fig. 1 and Sec. 5.1). The gains are even more significant at higher standards. To understand how far we are from human performance in 3D object detection, we have conducted a human label study to compare auto labels with human labels (Sec. 5.2). To our delight, we found that auto labels are already on par or even slightly better compared to human labels on the selected test segments.

In Sec. 5.3, we demonstrate the application of our pipeline for semi-supervised learning and show significantly improved student models trained with auto labels. We also conduct extensive ablation and analysis experiments to validate our design choices in Sec. 5.4 and Sec. 5.5 and provide visualization results in Sec. 5.6.

In summary, the contributions of our work are:

- Formulation of the offboard 3D object detection problem and proposal of a specific pipeline (3D Auto Labeling) that leverages our multi-frame detector and novel object-centric auto labeling models.
- State-of-the-art 3D object detection performance on the challenging Waymo Open Dataset.
- The human label study on 3D object detection with comparisons between human and auto labels.
- Demonstrated the effectiveness of auto labels for semi-supervised learning.

2. Related Work

3D object detection Most work has been focusing on using single-frame input. In terms of the representations used, they can be categorized into voxel-based [54, 10, 23, 51, 19, 63, 48, 71, 62, 20, 67, 55], point-based [46, 65, 37, 41, 64, 47], perspective-view-based [24, 35, 3] as well as hybrid strategy [70, 66, 7, 14, 45]. Several recent works explored temporal aggregation of Lidar scans for point cloud densification and shape completion. [32] fuses multi-frame information by concatenating feature maps from different frames. [15] aggregates (motion-compensated) points from

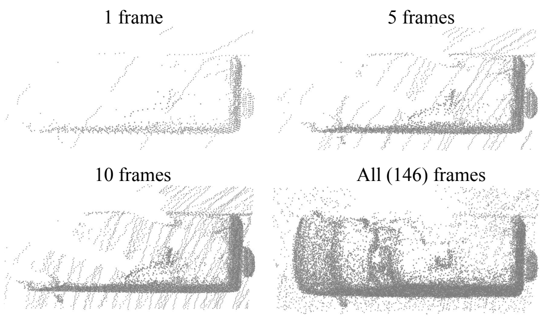

Figure 2. **Illustration of the complementary views of an object from the point cloud sequence.** Point clouds (aggregated from multiple frames) visualized in a top-down view for a mini-van.

different Lidar sweeps into a single scene. [68] uses graph-based spatiotemporal feature encoding to enable message passing among different frames. [16] encodes previous frames with a LSTM to assist detection in the current frame. Using multi-modal input (camera views and 3D point clouds) [19, 6, 42, 60, 27, 26, 34, 49, 40] has shown improved 3D detection performance compared to point-cloud-only methods, especially for small and far-away objects. In this work, we focus on a point-cloud-only solution and on leveraging data over a long temporal interval.

Learning from point cloud sequences Several recent works [30, 13, 36] proposed to learn to estimate scene flow from dynamic point clouds using end-to-end trained deep neural networks (from a pair of consecutive point clouds). Extending such ideas, MeteorNet [31] showed that longer sequences input can lead to performance gains for tasks such as action recognition, semantic segmentation and scene flow estimation. There are also other applications of learning in point cloud sequences, like point cloud completion [39], future point cloud prediction [57] and gesture recognition [38]. We also see more released datasets with sequence point cloud data such as the Waymo Open Dataset [52] for detection and the SemanticKITTI dataset [2] for 3D semantic segmentation.

Auto labeling The large datasets required for training data-hungry models have increased the annotation costs noticeably in recent years. Accurate auto labeling can dramatically reduce annotation time and cost. Previous works on auto labeling were mainly focused on 2D applications. Lee *et al.* proposed pseudo-labeling [21] to use the most confident predicted category of an image classifier as labels to train it on the unlabeled part of the dataset. More recent works [17, 72, 61, 59] have further improved the procedures to use pseudo labels and demonstrated wide success including state-of-the-art results on ImageNet [8].

For 3D object detection, recently, Zakharov *et al.* [69] proposed an auto labeling framework using pre-trained 2D

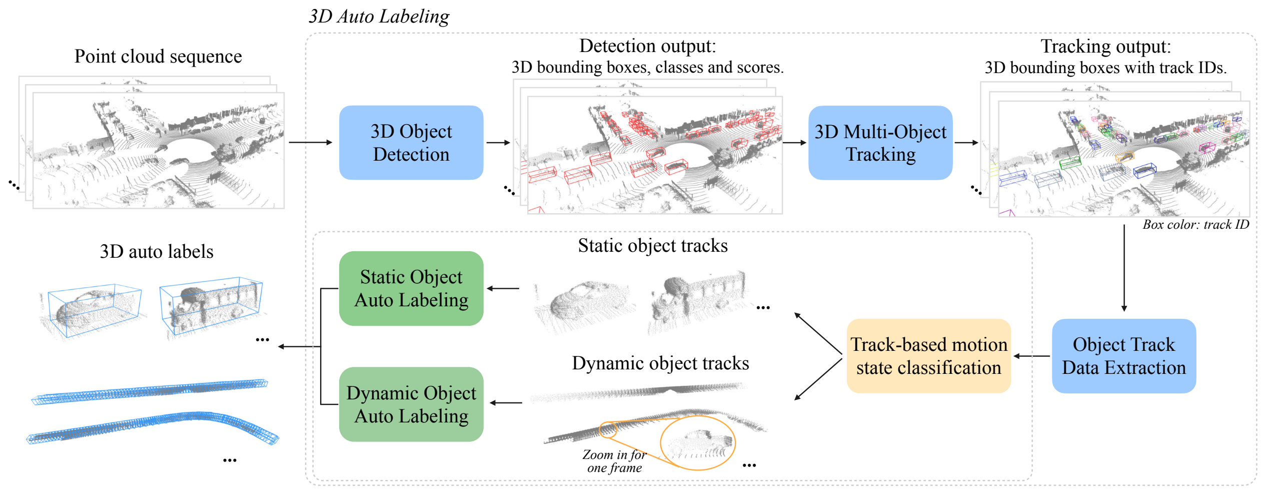

Figure 3. **The 3D Auto Labeling pipeline.** Given a point cloud sequence as input, the pipeline first leverages a 3D object detector to localize objects in each frame. Then object boxes at different frames are linked through a multi-object tracker. Object track data (its point clouds at every frame as well as its 3D bounding boxes) are extracted for each object and then go through the object-centric auto labeling (with a divide-and-conquer for static and dynamic tracks) to generate the final "auto labels", *i.e.* refined 3D bounding boxes.

detectors to annotate 3D objects. While effective for loose localization (*i.e.* IoU of 0.5), there is a considerable performance gap for applications requiring higher precision. [33] tried to leverage weak center-click supervision to reduce 3D labels needed. Several other works [5, 1, 22, 28, 11] have also proposed methods to assist human annotators and consequently reducing the annotation cost.

3. Offboard 3D Object Detection

Problem statement Given a sequence of sensor inputs (temporal data) of a dynamic environment, our goal is to localize and classify objects in the 3D scene for every frame. Specifically, we consider the input of a sequence of point clouds $\{\mathcal{P}_i \in \mathbf{R}^{n_i \times C}\}$, $i = 1, 2, ..., N$ with the point cloud $\mathcal{P}_i$ (n_i points with C channels for each point) of each of the N total frames. The point channels include the XYZ in the sensor's coordinate (at each frame) and other optional information such as color and intensity. We also assume known sensor poses $\{\mathcal{M}_i = [R_i|t_i] \in \mathbf{R}^{3\times4}\}$, $i = 1, 2, ..., N$ at each frame in the world coordinate, such that we can compensate the ego-motion. For each frame, we output amodal 3D bounding boxes (parameterized by its center, size and orientation), class types (*e.g.* vehicles) and unique object IDs for all objects that appear in the frame.

Design space Access to temporal data (history and future) has led to a much larger design space of detectors compared to just using single frame input.

One baseline design is to extend the single-frame 3D object detectors to use multi-frame input. Although previous works [32, 15, 16, 68] have shown its effectiveness, a multi-frame detector is hard to scale up to more than a few frames and cannot compensate the object motions since frame stacking is done for the entire scene. We observe that the contributions of multi-frame input to the detector quality diminish as we stack more frames (Table 5). Another idea is to extend the second stage of two-stage detectors [42, 46] to take object points from multiple frames. Compared to taking multi-frame input of the whole scene, the second-stage only processes proposed object regions. However, it is not intuitive to decide how many context frames to use. Setting a fixed number may work well for some objects but suboptimal for others.

Compared to the *frame-centric* designs above, where input is always from a fixed number of frames, we recognize the necessity to adaptively choose the temporal context size for each object independently, leading to an *object-centric* design. As shown in Fig. 3, we can leverage the powerful multi-frame detector to give us the initial object localizations. Then for each object, through tracking, we can extract all relevant object point clouds and detection boxes from all frames that it appears in. Subsequent models can take such object track data to output the final track-level refined boxes of the objects. As this process emulates how a human labeler annotates a 3D object in the point cloud sequence (localize, track and refine the track over time), we chose to refer to our pipeline as *3D Auto Labeling*.

4. 3D Auto Labeling Pipeline

Fig. 3 illustrates our proposed 3D Auto Labeling pipeline. We will introduce each module of the pipeline in the following sub-sections.

4.1. Multi-frame 3D Object Detection

MVF++ As the entry point to our pipeline, accurate object detection is essential for the downstream modules. In this work, we propose the MVF++ 3D detector by extending the top-performing Multi-View Fusion [70] (MVF) detector in three aspects: 1) to enhance the discriminative ability of point-level features, we add an auxiliary loss for 3D semantic segmentation, where points are labeled as positives/negatives if they lie inside/outside of a ground truth 3D box; 2) for obtaining more accurate training targets and improving training efficiency, we eliminate the anchor matching step in the MVF paper and adopt the anchor-free design as in [53]; 3) to leverage ample computational resources available in the offboard setting, we redesign the network architecture and increase the model capacity. Please see supplemental Sec. C for details.

Multi-frame MVF++ We extend the MVF model to use multiple LiDAR scans. Points from multiple consecutive scans are transformed to the current frame based on ego-motion. Each point is extended by one additional channel, encoding of the relative temporal offset, similar to [15]. The aggregated point cloud is used as the input to the MVF++.

Test-time augmentation We further boost the 3D detection through test-time augmentation (TTA) [18], by rotating the point cloud around Z-axis by 10 different angles (*i.e.* [0, $\pm 1/8\pi$, $\pm 1/4\pi$, $\pm 3/4\pi$, $\pm 7/8\pi$, π]), and ensembling predictions with weighted box fusion [50]. While it may lead to excessive computational complexity for onboard uses, in the offboard setting TTA can be parallelized across multiple devices for fast execution.

4.2. Multi-object Tracking

The multi-object tracking module links detected objects across frames. Given the powerful multi-frame detector, we choose to take the tracking-by-detection path and have a separate non-parametric tracker. This leads to a simpler and more modular design compared to the joint detection and tracking methods [32, 58, 25]. Our tracker is an implementation variant of the [56], using detector boxes for associations and Kalman filter for state updates.

4.3. Object Track Data Extraction

Given tracked detection boxes for an object, we can extract object-specific LiDAR point clouds from the sequence. We use the term *object track data* to refer to such 4D (3D spatial and 1D temporal) object information.

To extract object track data, we first transform all boxes and point clouds to the world coordinate through the known sensor poses to remove the ego-motion. For each unique object (according to the object ID), we crop its object points

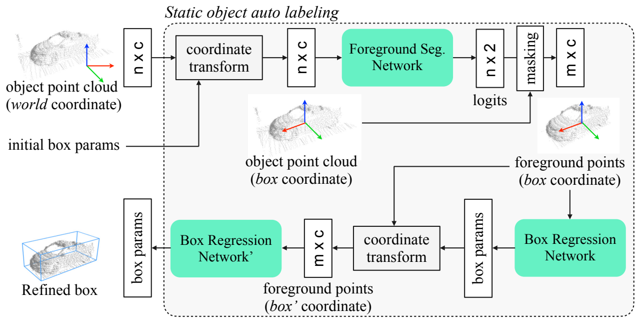

Figure 4. **The static object auto labeling model.** Taking as input the merged object points in the world coordinate, the model outputs a single box for the static object.

within the estimated detector boxes (enlarged by α meters in each direction to include more contexts). Such extraction gives us a sequence of object point clouds $\{\mathcal{P}_{j,k}\}$, $k \in S_j$ for each object j and its visible frames S_j. Fig. 3 visualizes the object points for several vehicles. Besides the raw point clouds, we also extract the tracked boxes for each object and every frame $\{\mathcal{B}_{j,k}\}$, $k \in S_j$ in the world coordinate.

4.4. Object-centric Auto Labeling

In this section, we describe how we take the object track data to "auto label" the objects. As illustrated in Fig. 3, the process includes three sub-modules: the track-based motion state classification, static object auto labeling and dynamic object auto labeling, which are described in detail below.

Divide and conquer: motion state estimation In the real world, lots of objects are completely static during a period of time. For example, parked cars or furniture in a room do not move within a few minutes or hours. In terms of offboard detection, it is preferred to assign a single 3D bounding box to a static object rather than separate boxes in different frames to avoid jittering.

Based on this observation, we take a divide-and-conquer approach to handle static and moving objects differently, introducing a module to classify an object's motion state (static or not) before the auto labeling. While it could be hard to predict an object's motion state from just a few frames (due to the perception noise), we find it relatively easy if all object track data is used. As the visualization in Fig. 3 shows, it is often obvious to tell whether an object is static or not from its trajectory. A linear classifier using a few heuristic features from the object track's boxes can already achieve 99%+ motion state classification accuracy for vehicles. More details are in supplemental Sec. E.

Static object auto labeling For a static object, the model takes the merged object point clouds ($\mathcal{P}_j = \cup\{\mathcal{P}_{j,k}\}$ in the world coordinate) from points at different frames and pre-

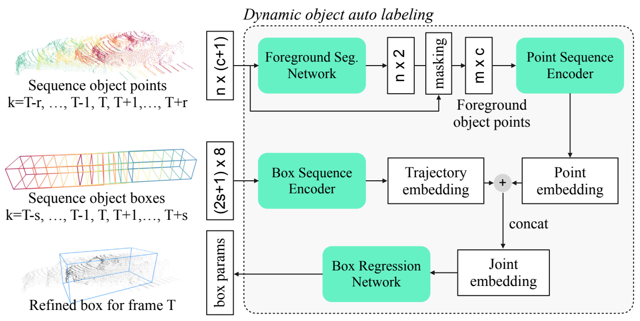

Figure 5. **The dynamic object auto labeling model.** Taking a sequence of object points and a sequence of object boxes, the model runs in a sliding window fashion and outputs a refined 3D box for the center frame. Input point and box colors represent frames.

dicts a single box. The box can then be transformed to each frame through the known sensor poses.

Fig. 4 illustrates our proposed model for static object auto labeling. Similar to [42, 46], we first transform (through rotation and translation) the object points to a *box coordinate* before the per-object processing, such that the point clouds are more aligned across objects. In the box coordinate, the $+X$ axis is the box heading direction, the origin is the box center. Since we have the complete sequence of the detector boxes, we have multiple options on which box to use as the initial box. The choice actually has a significant impact on model performance. Empirically, using the box with the highest detector score leads to the best performance (see supplemental Sec. I for an ablation).

To attend to the object, the object points are passed through an instance segmentation network to segment the foreground (m foreground points are extracted by the mask). Inspired by the Cascade-RCNN [4], we iteratively regress the object's bounding box. At test time, we can further improve box regression accuracy by test-time-augmentation (similar to Sec. 4.1).

All networks are based on the PointNet [43] architecture. The model is supervised by the segmentation and box estimation ground truths. Details of the architecture, losses and the training process are described in supplemental Sec. F.

Dynamic object auto labeling For a moving object, we need to predict different 3D bounding boxes for each frame. Due to the sequence input/output, the model design space is much larger than that for static objects. A baseline is to re-estimate the 3D bounding box with cropped point clouds. Similar to the smoothing in tracking, we can also refine boxes based on the sequence of the detector boxes. Another choice is to "align" or register object points with respect to a keyframe (*e.g.* the current frame) to obtain a denser point cloud for box estimation. However, the alignment can be a harder problem than box estimation especially for occluded or faraway objects with fewer points. Besides, it is challenging to align deformable objects like pedestrians.

We propose a design (Fig. 5) that leverages both the point cloud and the detector box sequences without aligning points to a keyframe explicitly. Given a sequence of object point clouds $\{\mathcal{P}_{j,k}\}$ and a sequence of detector boxes $\{\mathcal{B}_{j,k}\}$ for the object j at frames $k \in S_j$, the model predicts the object box at each frame k in a sliding window form. It consists of two branches, one taking the point sequence and the other taking the box sequence.

For the point cloud branch, the model takes a sub-sequence of the object point clouds $\{\mathcal{P}_{j,k}\}_{k=T-r}^{T+r}$. After adding a temporal encoding channel to each point (similar to [15]) , the sub-sequence points are merged through union and transformed to the box coordinate of the detector box $\mathcal{B}_{j,T}$ at the center frame. Following that, we have a PointNet [43] based segmentation network to classify the foreground points (of the $2r+1$ frames) and then encode the object points into an embedding through another point encoder network.

For the box sequence branch, the box sequences $\{\mathcal{B}'_{j,k}\}_{k=T-s}^{T+s}$ of $2s+1$ frames are transformed to the box coordinate of the detector box at frame T. Note that the box sub-sequence can be longer than the point sub-sequence to capture the longer trajectory shape. A box sequence encoder network (a PointNet variant) will then encode the box sequence into a trajectory embedding, where each box is a point with 7-dim geometry and 1-dim time encoding.

Next, the computed object embedding and the trajectory embedding are concatenated to form the joint embedding which will then be passed through a box regression network to predict the object box at frame T.

5. Experiments

We start the section by comparing our offboard 3D Auto Labeling with state-of-the-art 3D object detectors in Sec. 5.1. In Sec. 5.2 we compare the auto labels with the human labels. In Sec. 5.3, we show how the auto labels can be used to supervise a student model to achieve improved performance under low-label regime or in another domain. We provide analysis of the multi-frame detector in Sec. 5.4 and analysis experiments to validate our designs of the object-centric auto labeling models in Sec. 5.5 and finally visualize the results in Sec. 5.6.

Dataset We evaluate our approach using the challenging Waymo Open Dataset (WOD) [52], as it provides a large collection of LiDAR sequences, with 3D labels available for each frame. The dataset includes a total number of 1150 sequences with 798 for training, 202 for validation and 150 for testing. Each LiDAR sequence lasts around 20 seconds with a sampling frequency at 10Hz. For our experiments, we evaluate both 3D and bird's eye view (BEV) object detection metrics for vehicles and pedestrians.

Method	frames	*Vehicles* 3D AP IoU=0.7	3D AP IoU=0.8	BEV AP IoU=0.7	BEV AP IoU=0.8	*Pedestrians* 3D AP IoU=0.5	3D AP IoU=0.6	BEV AP IoU=0.5	BEV AP IoU=0.6
StarNet [37]	1	53.70	-	-	-	66.80	-	-	-
PointPillar [20]*	1	60.25	27.67	78.14	63.79	60.11	40.35	65.42	51.71
Multi-view fusion (MVF) [70]	1	62.93	-	80.40	-	65.33	-	74.38	-
AFDET [12]	1	63.69	-	-	-	-	-	-	-
ConvLSTM [16]	4	63.60	-	-	-	-	-	-	-
RCD [3]	1	68.95	-	82.09	-	-	-	-	-
PillarNet [55]	1	69.80	-	87.11	-	72.51	-	78.53	-
PV-RCNN [45]*	1	70.47	39.16	83.43	69.52	65.34	45.12	70.35	56.63
Single-frame MVF++ (Ours)	1	74.64	43.30	87.59	75.30	78.01	56.02	83.31	68.04
Multi-frame MVF++ w. TTA (Ours)	5	79.73	49.43	91.93	80.33	81.83	60.56	85.90	73.00
3D Auto Labeling (Ours)	all	**84.50**	**57.82**	**93.30**	**84.88**	**82.88**	**63.69**	**86.32**	**75.60**

Table 1. **3D object detection results for vehicles and pedestrians on the Waymo Open Dataset *val* set.** Methods in comparison include prior state-of-the-art single-frame based 3D detectors as well as our single-frame MVF++, our multi-frame MVF++ (5 frames) and our full 3D Auto Labeling pipeline. The metrics are L1 3D AP and bird's eye view (BEV) AP at two IoU thresholds: the common standard IoU=0.7 and a high standard IoU=0.8 for vehicles; and IoU = 0.5, 0.6 for pedestrians. * reproduced results using author's released code.

5.1. Comparing with State-of-the-art Detectors

In Table 1, we show comparisons of our 3D object detectors and the 3D Auto Labeling with various single-frame and multi-frame based detectors, under both the common standard IoU threshold and a higher standard IoU threshold to pressure test the models.

We show that our single-frame MVF++ has already outperformed the prior art single-frame detector PVR-CNN [45]. The multi-frame version of the MVF++, as a baseline of the offboard 3D detection methods, significantly improves upon the single-frame MVF++ thanks to the extra information from the context frames.

For vehicles, comparing the last three rows, our complete 3D Auto Labeling pipeline, which leverages the multi-frame MVF++ and the object-centric auto labeling models, further improves the detection quality especially in the higher standard at IoU threshold of 0.8. It improves the 3D AP@0.8 significantly by **14.52** points compared to the single-frame MVF++ and by **8.39** points compared to the multi-frame MVF++, which is already very powerful by itself. These results show the great potential of leveraging the long sequences of point clouds for offboard perception.

We also show the detection AP for the pedestrian class, where we consistently observe the leading performance of the 3D Auto Labeling pipeline especially at the higher localization standard (IoU=0.6) with **7.67** points gain compared to the single-frame MVF++ and **3.13** points gain compared to the multi-frame MVF++.

5.2. Comparing with Human Labels

In many perception domains such as image classification and speech recognition, researchers have collected data to understand humans' capability [44, 9, 29]. However, to the best of our knowledge, no such study exists for 3D recognition especially for 3D object detection. To fill this gap, we conducted a small-scale human label study on the Waymo Open Dataset to understand the capability of human in recognizing objects in a dynamic 3D scene. We randomly selected 5 sequences from the Waymo Open Dataset *val* set and asked three experienced labelers to re-label each sequence independently (with the same labeling protocol as WOD).

	3D AP IoU=0.7	3D AP IoU=0.8	BEV AP IoU=0.7	BEV AP IoU=0.8
Human	**86.45**	**60.49**	**93.86**	86.27
3DAL (Ours)	85.37	56.93	92.80	**87.55**

Table 2. **Comparing human labels and auto labels in 3D object detection.** The metrics are 3D and BEV APs for vehicles on the 5 sequences from the Waymo Open Dataset *val* set. Human APs are computed by comparing them with the WOD's released ground truth and using number of points in boxes as human label scores.

In Table 2, we report the mean AP of human labels and auto labels across the 5 sequences. With the common 3D AP@0.7 (L1) metric, the auto labels are only around 1 point lower than the average labeler, although the gap is slightly larger in the more strict 3D AP@0.8 metric. With some visualization, we found the larger gap is mostly caused by inaccurate heights. The comparisons with the BEV AP@0.8 metric verifies our observation: when we don't consider height, the auto labels even outperform the average human labels by 1.28 points.

With such high quality, we believe the auto labels can be used to pre-label point cloud sequences to assist and accelerate human labeling, or be used directly to train lightweight student models as shown in the following section.

Training Data	Test Data	3D AP	BEV AP
100% main *train* (Human)	main *val*	71.2	86.9
10% main *train* (Human)	main *val*	64.3	81.2
10% main *train* (Human) + 90% main *train* (**3DAL**)	main *val*	70.0	86.4
100 % main *train* (Human)	domain *test*	59.4	*N/A*
100 % main *train* (Human) + domain (Self Anno.)	domain *test*	60.3	*N/A*
100 % *train* (Human) + domain (**3DAL**)	domain *test*	64.2	*N/A*

Table 3. **Results of semi-supervised learning with auto labels.** Metrics are 3D and BEV AP for vehicles on the Waymo Open Dataset. The type of annotation is reported in parenthesis. Please note, test set BEV AP is not provided by the submission server.

anchor-free	cap. increase	seg loss	5-frame	TTA	AP@0.7/0.8
✓	-	-	-	-	71.20 / 39.70
✓	✓	-	-	-	74.28 / 42.91
✓	✓	✓	-	-	74.64 / 43.30
✓	✓	✓	✓	-	76.34 / 45.57
✓	✓	✓	✓	✓	79.73 / 49.43

Table 4. **Ablation studies on the improvements to 3D detector MVF** [70]. Metrics are 3D AP (L1) at IoU thresholds 0.7 and 0.8 for vehicles on the Waymo Open Dataset *val* set.

# frames	1	2	3	4	5	10
AP@0.7	74.64	75.32	75.63	76.17	76.34	76.96
AP@0.8	43.30	44.11	44.80	45.43	45.57	46.20

Table 5. **Ablation studies on 3D detection AP *vs*. temporal contexts.** Metrics are 3D AP (L1) for vehicles on the Waymo Open Dataset *val* set. We used the 5-frame model in 3D Auto Labeling.

5.3. Applications to Semi-supervised Learning

In this section, we study the effectiveness of our auto labeling pipeline in the task of semi-supervised learning to train a student model under two settings: intra-domain and cross-domain. We choose the student model as a single-frame MVF++ detector that can run in real-time.

For the *intra-domain semi-supervised learning*, we randomly select 10% sequences (79 ones) in the main WOD training set to train our 3D Auto Labeling (3DAL) pipeline. Once trained, we apply it to the rest 90% sequences (719 ones) in the main training set to generate "auto labels" (we only keep boxes with scores higher than 0.1). In Table 3 (first two rows), we see that reducing the human annotations to 10% significantly lowers the student model's performance. However, when we use auto labels, the student model trained on 10% human labels and 90% auto labels can get similar performance compared to using 100% human labels (AP gaps smaller than 1 point), demonstrating superb data efficiency auto labels can provide.

For the *cross-domain semi-supervised learning*, the teacher auto labels data from an unseen domain. The teacher is trained on the main WOD *train* set, and auto labels the domain adaptation WOD *train* and *unlabeled* sets (separate 680 sequences from the main WOD). The student is then trained on the union of these three sets. Evaluations are on the domain adaptation *test* set. The last three rows of Table 3 show the results. Without using any data from the new domain, the student gets an AP of 59.4. While using the student to self-label slightly helps (improves the results by ~ 1 point), using our 3DAL to auto label the new domain significantly improves the student AP by ~ 5 points.

5.4. Analysis of the Multi-frame Detector

Table 4 shows the ablations of our proposed MVF++ detectors. We see that the offboard techniques such as the model capacity increase (+3.08 AP@0.7), using point clouds from 5 frames as input (+1.70 AP@0.7) and test time augmentation (+3.39 AP@0.7) are all very effective in improving the detection quality.

Table 5 shows how the number of consecutive input frames impacts the detection APs. The gains of adding frames quickly diminishes as the number of frames increases: *e.g.* while the AP@0.8 improves by 0.81 from 1 to 2 frames, the gain from 4 to 5 frames is only 0.14 point.

5.5. Analysis of Object Auto Labeling Models

We evaluate the object auto labeling models using the box accuracy metric under two IoU thresholds 0.7 and 0.8 on the Waymo Open Dataset *val* set. The predicted box is considered correct if its IoU with the ground truth is higher than the threshold. More analysis is in supplemental Sec. I.

Ablations of the static object auto labeling In table 6 we can see the importance of the initial coordinate transform (to the box coordinate), and the foreground segmentation network in the first 3 rows. In the 4th and the 5th rows, we see the gains of using iterative box re-estimation and test time augmentation respectively.

Alternative designs of the dynamic object auto labeling Table 7 ablates the design of the dynamic object auto labeling model. For the align & refine model, we use the multi-frame MVF++ detector boxes to "align" the object point clouds from the nearby frames ($[-2, +2]$) to the center frame. For each context frame, we transform the coordinate by aligning the center and heading of the context frame boxes to the center frame box. The model using unaligned point clouds (in the center frame's coordinate, from $[-2, +2]$ context frames), second row, actually gets higher accuracy (second row) than the aligned one. The model taking only the box sequence (third row) as input performs reasonably as well, by leveraging the trajectory shape and the box sizes. Our model jointly using the multi-frame object point clouds and the box sequences gets the best accuracy.

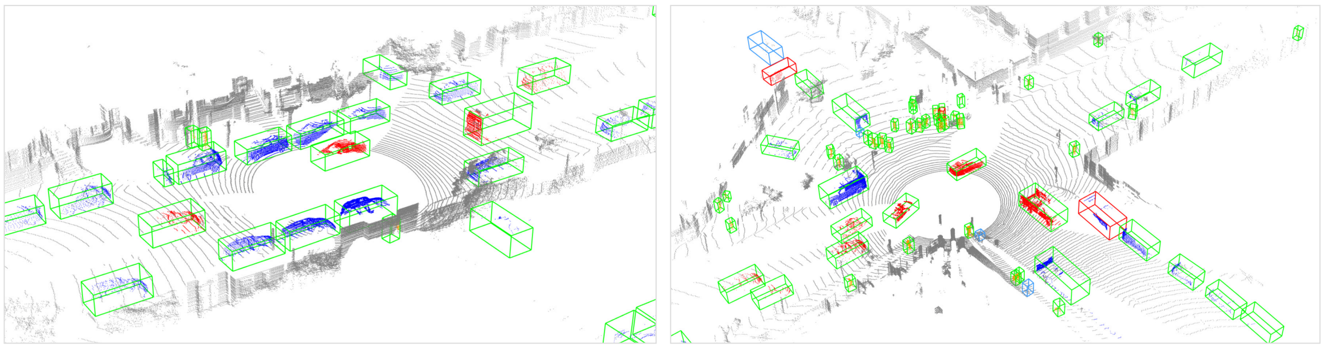
Figure 6. **Visualization of 3D auto labels on the Waymo Open Dataset *val* set** (best viewed in color with zoom in). Object points are colored by object types with blue for static vehicles, red for moving vehicles and orange for pedestrians. Boxes are colored as: green for true positive detections, red for false positives and cyan for ground truth boxes in the cases of false negatives.

transform	segmentation	iterative	tta	Acc@0.7/0.8
-	-	-	-	78.82 / 50.90
✓	-	-	-	81.35 / 54.76
✓	✓	-	-	81.37 / 55.67
✓	✓	✓	-	82.02 / 56.77
✓	✓	✓	✓	**82.28 / 56.92**

Table 6. **Ablation studies of the static auto labeling model.** Metrics are the box accuracy at 3D IoU=0.7 and IoU=0.8 for vehicles in the Waymo Open Dataset *val* set.

Method	Acc@0.7/0.8
Align & refine	83.33 / 60.69
Points only	83.79 / 61.95
Box sequence only	83.13 / 58.96
Points and box sequence joint	**85.67 / 65.77**

Table 7. **Comparing with alternative designs of dynamic object auto labeling.** Metrics are box accuracy with 3D IoU thresholds 0.7 and 0.8 for vehicles on the Waymo Open Dataset *val* set.

Effects of temporal context sizes for object auto labeling Table 8 studies how the context frame sizes influence the box prediction accuracy. We also compare with our single-frame (S-MVF++) and multi-frame detectors (M-MVF++) to show extra gains the object auto labeling can bring. We can clearly see that using large temporal contexts improves the performance while using the entire object track (the last row) leads to the best performance. Note that for the static object model, we use the detector box with the highest score for the initial coordinate transform, which gives our auto labeling an advantage over frame-based method.

5.6. Qualitative Analysis

In Fig. 6, we visualize the auto labels for two representative scenes in autonomous driving: driving on a road with parked cars, and passing a busy intersection. Our model is able to accurately recognize vehicles and pedestrians in challenging cases with occlusions and very few points. The busy intersection scene also shows a few failure cases including false negatives of pedestrians in rare poses (sitting), false negatives of severely occluded objects and false positive for objects with similar geometry to cars. Those hard cases can potentially be solved with added camera information with multi-modal learning.

Method	Context frames	*static* Acc@0.7/0.8	*dynamic* Acc@0.7/0.8
S-MVF++	$[-0, +0]$	67.17 / 36.61	80.07 / 57.71
M-MVF++	$[-4, +0]$	73.96 / 43.56	82.21 / 59.52
3DAL	$[-0, +0]$	78.13 / 50.30	80.65 / 57.97
	$[-2, +2]$	79.60 / 52.52	84.34 / 63.60
	$[-5, +5]$	80.48 / 55.02	85.10 / 64.51
	all	**82.28 / 56.92**	**85.67 / 65.77**

Table 8. **Effects of temporal context sizes for object auto labeling.** Metrics are the box accuracy at 3D IoU=0.7, 0.8 for vehicles in the WOD *val* set. Dynamic vehicles have a higher accuracy because they are closer to the sensor than static ones.

6. Conclusion

In this work we have introduced 3D Auto Labeling, a state-of-the-art offboard 3D object detection solution using point cloud sequences as input. The pipeline leverages the long-term temporal data of objects in the 3D scene. Key to our success are our object-centric formulation, powerful offboard multi-frame detector and novel object auto labeling models. Evaluated on the Waymo Open Dataset, our solution has shown significant gains over prior art onboard 3D detectors, especially with high standard metrics. A human label study has further shown the high quality of the auto labels reaching comparable performance as experienced humans. Moreover, the semi-supervised learning experiments have demonstrated the usefulness of the auto labels for student training in cases of low-label and unseen domains.

References

[1] David Acuna, Huan Ling, Amlan Kar, and Sanja Fidler. Efficient interactive annotation of segmentation datasets with polygon-rnn++. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 859–868, 2018.

[2] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Jurgen Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9297–9307, 2019.

[3] Alex Bewley, Pei Sun, Thomas Mensink, Dragomir Anguelov, and Cristian Sminchisescu. Range conditioned dilated convolutions for scale invariant 3d object detection, 2020.

[4] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *CVPR*, 2018.

[5] Lluis Castrejon, Kaustav Kundu, Raquel Urtasun, and Sanja Fidler. Annotating object instances with a polygon-rnn. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5230–5238, 2017.

[6] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *CVPR*, 2017.

[7] Y. Chen, S. Liu, X. Shen, and J. Jia. Fast point r-cnn. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9774–9783, 2019.

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*. IEEE, 2009.

[9] Neeraj Deshmukh, Richard Jennings Duncan, Aravind Ganapathiraju, and Joseph Picone. Benchmarking human performance for continuous speech recognition. In *Proceeding of Fourth International Conference on Spoken Language Processing. ICSLP'96*, volume 4, pages 2486–2489. IEEE, 1996.

[10] M. Engelcke, D. Rao, D. Z. Wang, C. H. Tong, and I. Posner. Vote3deep: Fast object detection in 3d point clouds using efficient convolutional neural networks. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1355–1361, May 2017.

[11] Di Feng, Xiao Wei, Lars Rosenbaum, Atsuto Maki, and Klaus Dietmayer. Deep active learning for efficient training of a lidar 3d object detector. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 667–674. IEEE, 2019.

[12] Runzhou Ge, Zhuangzhuang Ding, Yihan Hu, Yu Wang, Sijia Chen, Li Huang, and Yuan Li. Afdet: Anchor free one stage 3d object detection, 2020.

[13] Xiuye Gu, Yijie Wang, Chongruo Wu, Yong Jae Lee, and Panqu Wang. Hplflownet: Hierarchical permutohedral lattice flownet for scene flow estimation on large-scale point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3254–3263, 2019.

[14] 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 (CVPR)*, June 2020.

[15] Peiyun Hu, Jason Ziglar, David Held, and Deva Ramanan. What you see is what you get: Exploiting visibility for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[16] Rui Huang, Wanyue Zhang, Abhijit Kundu, Caroline Pantofaru, David A. Ross, Thomas A. Funkhouser, and Alireza Fathi. An LSTM approach to temporal 3d object detection in lidar point clouds. *CoRR*, 2020.

[17] Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondrej Chum. Label propagation for deep semi-supervised learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5070–5079, 2019.

[18] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90, May 2017.

[19] 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.

[20] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *CVPR*, 2019.

[21] Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, 2013.

[22] Jungwook Lee, Sean Walsh, Ali Harakeh, and Steven L Waslander. Leveraging pre-trained 3d object detection models for fast ground truth generation. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 2504–2510. IEEE, 2018.

[23] B. 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, Sep. 2017.

[24] Bo Li, Tianlei Zhang, and Tian Xia. Vehicle detection from 3d lidar using fully convolutional network. In *RSS 2016*, 2016.

[25] Peiliang Li, Jieqi Shi, and Shaojie Shen. Joint spatial-temporal optimization for stereo 3d object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6877–6886, 2020.

[26] M. Liang, B. Yang, Y. Chen, R. Hu, and R. Urtasun. Multi-task multi-sensor fusion for 3d object detection. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7337–7345, 2019.

[27] Ming Liang, Bin Yang, Shenlong Wang, and Raquel Urtasun. Deep continuous fusion for multi-sensor 3d object detection. In *ECCV*, 2018.

[28] Huan Ling, Jun Gao, Amlan Kar, Wenzheng Chen, and Sanja Fidler. Fast interactive object annotation with curve-gcn. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5257–5266, 2019.

[29] Richard P Lippmann. Speech recognition by machines and humans. *Speech communication*, 22(1):1–15, 1997.

[30] Xingyu Liu, Charles R Qi, and Leonidas J Guibas. Flownet3d: Learning scene flow in 3d point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 529–537, 2019.

[31] Xingyu Liu, Mengyuan Yan, and Jeannette Bohg. Meteornet: Deep learning on dynamic 3d point cloud sequences. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9246–9255, 2019.

[32] 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 (CVPR)*, June 2018.

[33] Qinghao Meng, Wenguan Wang, Tianfei Zhou, Jianbing Shen, Luc Van Gool, and Dengxin Dai. Weakly supervised 3d object detection from lidar point cloud. *arXiv preprint arXiv:2007.11901*, 2020.

[34] G. P. Meyer, J. Charland, D. Hegde, A. Laddha, and C. Vallespi-Gonzalez. Sensor fusion for joint 3d object detection and semantic segmentation. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1230–1237, 2019.

[35] Gregory P. Meyer, Ankit Laddha, Eric Kee, Carlos Vallespi-Gonzalez, and Carl K. Wellington. LaserNet: An efficient probabilistic 3D object detector for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[36] Himangi Mittal, Brian Okorn, and David Held. Just go with the flow: Self-supervised scene flow estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11177–11185, 2020.

[37] Jiquan Ngiam, Benjamin Caine, Wei Han, Brandon Yang, Yuning Chai, Pei Sun, Yin Zhou, Xi Yi, Ouais Alsharif, Patrick Nguyen, Zhifeng Chen, Jonathon Shlens, and Vijay Vasudevan. Starnet: Targeted computation for object detection in point clouds. *CoRR*, 2019.

[38] Joshua Owoyemi and Koichi Hashimoto. Spatiotemporal learning of dynamic gestures from 3d point cloud data. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–5. IEEE, 2018.

[39] Lukas Prantl, Nuttapong Chentanez, Stefan Jeschke, and Nils Thuerey. Tranquil clouds: Neural networks for learning temporally coherent features in point clouds. *arXiv preprint arXiv:1907.05279*, 2019.

[40] Charles R Qi, Xinlei Chen, Or Litany, and Leonidas J Guibas. Imvotenet: Boosting 3d object detection in point clouds with image votes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4404–4413, 2020.

[41] 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 International Conference on Computer Vision*, pages 9277–9286, 2019.

[42] Charles R Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. Frustum pointnets for 3d object detection from rgb-d data. In *CVPR*, 2018.

[43] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *CVPR*, 2017.

[44] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[45] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10529–10538, 2020.

[46] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. *arXiv preprint arXiv:1812.04244*, 2018.

[47] Weijing Shi and Ragunathan (Raj) Rajkumar. Point-gnn: Graph neural network for 3d object detection in a point cloud. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[48] 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.

[49] Vishwanath A. Sindagi, Yin Zhou, and Oncel Tuzel. Mvx-net: Multimodal voxelnet for 3d object detection. In *International Conference on Robotics and Automation, ICRA 2019, Montreal, QC, Canada, May 20-24, 2019*, pages 7276–7282. IEEE, 2019.

[50] Roman Solovyev, Weimin Wang, and Tatiana Gabruseva. Weighted boxes fusion: ensembling boxes for object detection models. *arXiv preprint arXiv:1910.13302*, 2019.

[51] Shuran Song and Jianxiong Xiao. Deep sliding shapes for amodal 3d object detection in rgb-d images. In *CVPR*, 2016.

[52] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2446–2454, 2020.

[53] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. FCOS: Fully convolutional one-stage object detection. In *Proc. Int. Conf. Computer Vision (ICCV)*, 2019.

[54] Dominic Zeng Wang and Ingmar Posner. Voting for voting in online point cloud object detection. In *Proceedings of Robotics: Science and Systems*, Rome, Italy, July 2015.

[55] Yue Wang, Alireza Fathi, Abhijit Kundu, David Ross, Caroline Pantofaru, Thomas Funkhouser, and Justin Solomon. Pillar-based object detection for autonomous driving. In *ECCV*, 2020.

[56] Xinshuo Weng and Kris Kitani. A baseline for 3d multi-object tracking. *arXiv preprint arXiv:1907.03961*, 2019.

[57] Xinshuo Weng, Jianren Wang, Sergey Levine, Kris Kitani, and Nicholas Rhinehart. 4d forecasting: Sequential forecasting of 100,000 points, 2020.

[58] Xinshuo Weng, Ye Yuan, and Kris Kitani. Joint 3d tracking and forecasting with graph neural network and diversity sampling. *arXiv preprint arXiv:2003.07847*, 2020.

[59] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10687–10698, 2020.

[60] Danfei Xu, Dragomir Anguelov, and Ashesh Jain. PointFusion: Deep sensor fusion for 3d bounding box estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[61] I Zeki Yalniz, Hervé Jégou, Kan Chen, Manohar Paluri, and Dhruv Mahajan. Billion-scale semi-supervised learning for image classification. *arXiv preprint arXiv:1905.00546*, 2019.

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

[63] 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.

[64] 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.

[65] Zetong Yang, Yanan Sun, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Ipod: Intensive point-based object detector for point cloud, 2018.

[66] Z. Yang, Y. Sun, S. Liu, X. Shen, and J. Jia. Std: Sparse-to-dense 3d object detector for point cloud. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1951–1960, 2019.

[67] M. Ye, S. Xu, and T. Cao. Hvnet: Hybrid voxel network for lidar based 3d object detection. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1628–1637, 2020.

[68] J. Yin, J. Shen, C. Guan, D. Zhou, and R. Yang. Lidar-based online 3d video object detection with graph-based message passing and spatiotemporal transformer attention. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11492–11501, 2020.

[69] Sergey Zakharov, Wadim Kehl, Arjun Bhargava, and Adrien Gaidon. Autolabeling 3d objects with differentiable rendering of sdf shape priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12224–12233, 2020.

[70] Yin Zhou, Pei Sun, Yu Zhang, Dragomir Anguelov, Jiyang Gao, Tom Ouyang, James Guo, Jiquan Ngiam, and Vijay Vasudevan. End-to-end multi-view fusion for 3d object detection in lidar point clouds. In *Conference on Robot Learning*, pages 923–932, 2020.

[71] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *CVPR*, 2018.

[72] Yang Zou, Zhiding Yu, Xiaofeng Liu, BVK Kumar, and Jinsong Wang. Confidence regularized self-training. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5982–5991, 2019.