TCC-Det: Temporarily consistent cues for weakly-supervised 3D detection

Jan Skvrna and Lukas Neumann

Visual Recognition Group, Department of Cybernetics
FEE, Czech Technical University, Prague, Czech Republic

Abstract. Accurate object detection in LiDAR point clouds is a key prerequisite of robust and safe autonomous driving and robotics applications. Training the 3D object detectors currently involves the need to manually annotate vasts amounts of training data, which is very time-consuming and costly. As a result, the amount of annotated training data readily available is limited, and moreover these annotated datasets likely do not contain edge-case or otherwise rare instances, simply because the probability of them occurring in such a small dataset is low.
In this paper, we propose a method to train 3D object detector without any need for manual annotations, by exploiting existing off-the-shelf vision components and by using the consistency of the world around us. The method can therefore be used to train a 3D detector by only collecting sensor recordings in the real world, which is extremely cheap and allows training using orders of magnitude more data than traditional fully-supervised methods.
The method is evaluated on KITTI and Waymo Open datasets, where it outperforms all previous weakly-supervised methods and where it narrows the gap when compared to methods using human 3D labels.
The source code of our method is publicly available at `https://www.github.com/jskvrna/TCC-Det`.

Keywords: 3D Detection · Weakly-supervised · Autonomous driving

1 Introduction

Accurate object detection in LiDAR point clouds is a critical component of many applications, ranging from robotics to autonomous driving. One of the limitations that currently hinders exploitation of 3D object detectors in real-world scenarios is the scarcity of labelled training data, owing to the fact that human labelling in 3D is very time-consuming and therefore costly, as labelling one object instance can take up to 100 seconds [7, 25]. Training data are also specific to given country [28], and it is not practicable to assume there is a labelled dataset for every country. On the other hand, vast amounts of data are readily available, because capturing and storing sensor data is relatively cheap; the data are just not annotated and therefore useless for traditional fully-supervised 3D detection methods.

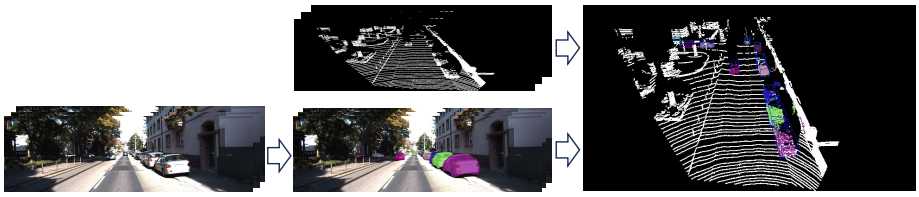

Fig. 1: Combining raw unlabelled RGB camera and LiDAR sensor data across multiple frames in a temporally consistent manner allows us to exploit a generic off-the-shelf 2D object detector to train a 3D object (vehicle) detector for LiDAR point clouds.

In our method, we aim to narrow this gap by training a standard 3D object detector, but **without using any human labels** in the process, therefore allowing the detector to be trained using the large quantities of unlabelled data readily available. Instead, we exploit an off-the-shelf 2D detector for the RGB camera (trained on a generic non-related dataset such as MS COCO [10]) and a number of real-world priors such as a generic shape of a car or the fact other objects only move subject to constraints given by the laws of physics between individual frames to train the detector. The result of our training process is a traditional 3D object detector that operates on LiDAR point clouds; the only yet crucial difference is the training signal (the training loss) used for the training, which does not rely on human annotations.

In this paper, we make the following contributions: (i) We introduce a Template Fitting Loss that is a relaxed formulation of Chamfer distance loss which is able to better estimate object location in noisy LiDAR scans and which takes into account multiple shape hypotheses at the same time. (ii) Thanks to velocity motion model for surrounding vehicles, we exploit the fact the same object is captured in subsequent LiDAR scans and use this temporal consistency to get more robust training signal for the network (iii) A novel Apperance Mask Loss is introduced to ensure the object detection in the LiDAR point cloud is consistent with the same object which is captured in the camera image.

The rest of the paper is structured as follows. In Section 2, an overview of prior work is presented. In Section 3, our proposed method is described and in Section 4 extensive experimental validation including numerous ablation experiments is presented. The paper is concluded in Section 5.

2 Related work

Fully supervised methods. PointNet [18] utilizes max pooling on the extracted features from points to learn to select interesting points and then use them in the fully connected layers to generate predictions. PointPillars [9] encodes point clouds into vertical columns (pillars), which allows the use of the 2D convolution, as the data is in Bird's Eye View (BEV). PV-RCNN [23] combines 3D sparse voxel convolutions and PointNet-based networks, thus combining grid-based and point-based methods to aggregate advances of both methods. Voxel-RCNN [2]

consists of a 3D sparse voxel convolution network, whose output is used both in 2D BEV Region proposal network (RPN) to get coarse predictions and in the Detection head, which uses Voxel ROI pooling, to get the fine refinement of the predictions provided by the 2D BEV RPN. The key advantage of grid-based methods (PointPillars, Voxel-RCNN) is fast inference speed and training compared to point-based methods (PointNet). CasA [30] adapts the 2D object detection cascade models to 3D as it uses a region proposal network accompanied by a cascade refinement network.

Weakly supervised methods. VS3D [19] detects regions of interest by the normalized point density in the LiDAR point cloud. To train the 3D bounding boxes prediction layer, it leverages the information of 2D bounding boxes detected by an off-the-shelf detector. Zakharov et al. [33] use an off-the-shelf 2D detector with a novel differentiable renderer of a DeepSDF framework [17] to auto-label 3D bounding boxes of cars. The method was pre-trained on the synthetic dataset and then trained on the actual dataset while iteratively adding more complex samples. McCraith et al. [14] use an off-the-shelf 2D detector accompanied by direct optimization of a template mesh to the LiDAR point clouds. The method showed that proper handling of the outliers is key to achieving good accuracy.

FGR [29] uses 2D ground truth bounding boxes and sparse LiDAR point cloud as an input. The method first uses a 3D coarse segmentation stage with RANSAC [3] to filter out the ground plane. The second stage performs the 3D bounding box estimation with context-aware adaptive region growing, key vertex localization and frustum intersection. The method was used to train PointR-CNN [24]. It is worth noting that precise amodal 2D ground truth bounding boxes are needed for this method.

WS3Dv2 [15] employs BEV click-point human annotations, representing the object's centre. The first stage generates the BEV points of interest, while the second stage learns to generate 3D bounding boxes at those points of interest. 500 frames with 534 precisely labelled objects are used to train both stages. The method's performance is close to fully supervised trained PointPillars [9] and PointRCNN [24] while being trained with less human-annotated data as those fully supervised ones were trained on 3712 frames.

MAP-Gen [12] and MTrans [11] use the same 3D training data as WS3Dv2 [15]. Both provide a way to generate new 3D points out of a 2D RGB image to deal with the problem of the LiDAR data sparsity. MTrans further closes the gap and achieves state-of-the-art performance.

Synthetic data. Another approach to address the lack of human annotations is using synthetic data, where annotations are generated alongside the synthetic data themselves. There are many synthetic driving datasets that have been proposed, such as GTA-V [20], Virtual KITTI [4], SYNTHIA [21], SHIFT [27], OPV2V [32] or most recently GAIA-1 [6]. The fidelity and data volumes of these datasets are growing constantly, but we believe that they still cannot fully replace data capture, especially to capture long-tail events, i.e. events which happen very rarely, and methods that allow training on unlabelled data, such as ours, are going to be complementary to the synthetic data approach.

3 Method

In traditional 3D object detection methods, training signal comes from (dis)agreement of network predictions with human annotations, which have to be manually created for each scene. In our method, human annotations are replaced by pseudo ground truth labels and further supported by two additional training signals which ensure network predictions are consistent with approximate 3D object shape in LiDAR pointcloud as well as with appearance of the same object in an RGB camera. We demonstrate that these pseudo ground truth labels together with two additional training signals, implemented as two individual loss functions – Template Fitting Loss and Appearance Mask Loss – are sufficient to train a standard 3D object detector from scratch without using any human annotations.

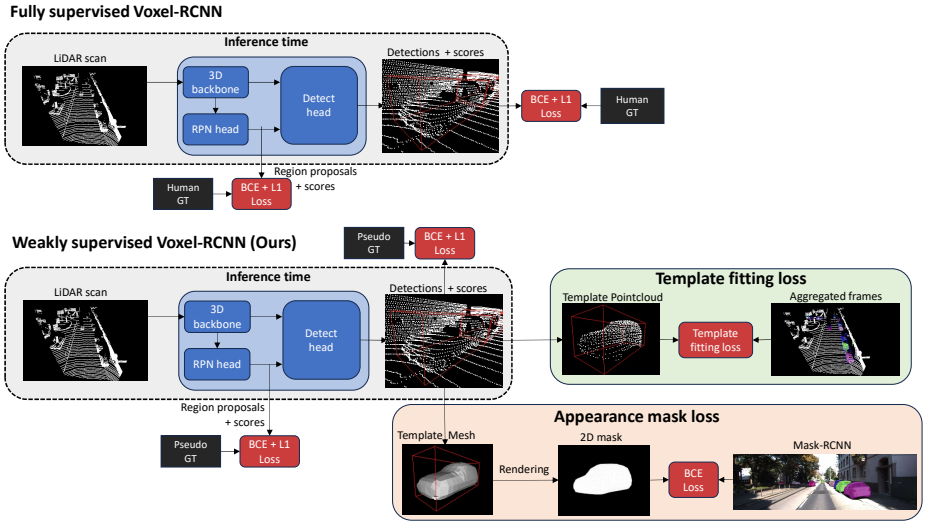

Fig. 2: Training pipelines of the traditional fully-supervised 3D object detector relying on 3D human annotations (top) and of the proposed method relying of 2D detections and shape prior hypotheses (bottom).

3.1 Template Fitting Loss (TFL)

The first loss function ensures that network 3D predictions can be "explained" by matching a rigid shape model of a vehicle to the predicted location. In other words, if the network predicts that there is a car at certain location, there should be a sufficient number of LiDAR points in that location of the world to support that, with an overall shape resembling to a car (see Figure 3).

The *Template Fitting Loss* exploits this observation by calculating the degree of agreement between the observed LiDAR points L_i for a given vehicle i and

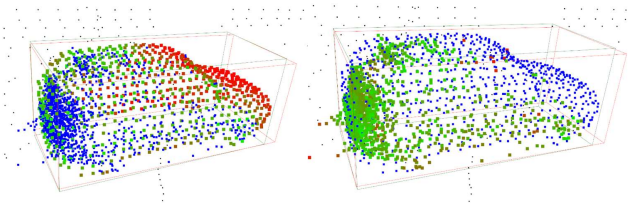

Fig. 3: Template Fitting Loss (TFL) sums up the distance of every shape model point to the nearest LiDAR detection (left) with the distance of every LiDAR detection to the nearest shape model point (right) in a symmetrical fashion, while discarding potential outliers. Green to red encodes low to high distance value.

the shape model M which is placed to the predicted position X_i, Y_i, Z_i with yaw θ_i, height H_i and scale (length/width) S_i

$$\mathcal{L}_{\text{TFL}}(L_i, P_i, M) = D(L_i, M \otimes P_i) + D(M \otimes P_i, L_i)$$
$$D(A, B) = \frac{1}{|A|} \sum_{a \in A} \sigma(k \cdot \min_{b \in B} \|a - b\|_2^2) \tag{1}$$

where $P_i = (X_i, Y_i, Z_i, \theta_i, H_i, S_i) \in \mathbb{R}^6$ denotes the network output (prediction), $M \otimes P_i$ denotes the shape model M transformed to the position $(X_i.Y_i, Z_i)$, rotated by θ_i and scaled by H_i and S_i, $\sigma(x)$ is the sigmoid activation function and k is a steepness parameter whose value is determined empirically. We note that S_i encodes both the width and length of the vehicle, as we believe long cars are wide and vice versa, and coupling width and height together through a single scaling factor achieved better accuracy.

The loss in Eq. (1) is a modified Chamfer distance loss, which can accommodate outliers and it permits for the fact that the two point clouds are not exactly the same, which is key because every car is different, and we cannot expect a perfect fit as our shape models $M \in \mathcal{M}$ are fairly generic.

Object Point Cloud. The LiDAR point cloud L_i for each object (vehicle) i is created in an offline pre-processing step, where an off-the-shelf instance segmentation method [31] which already had been trained on a generic dataset such as MS COCO [10] is used to detect objects (vehicles) and their instance segmentation B_i in the RGB camera image. Next, whole LiDAR scan is projected into the camera coordinate system and individual LiDAR points are matched to individual 2D segmentation masks in the camera image or discarded as background.

Temporal Consistency. LiDAR scans on one end tend to be sparse, on the other hand they may contain many outliers which are incorrectly associated with the object point cloud L_i, typically because of imperfect instance segmentation or "see-through" surfaces. Using data from a single LiDAR scan in the loss function leads to inferior results, especially for cars which are further away (see Sec. 4.5).

To address this issue, we exploit the availability of LiDAR scans and video sequence captured before and after the reference frame. We track all vehicles detected in the reference frames across the sequence by estimating their 3D location as the median of the object point cloud and then assigning detections

to the nearest vehicle in the subsequent frame, measured as the distance in the 3D world. This allows us to aggregate LiDAR points corresponding to the same car across multiple LiDAR scans, thus creating a more dense LiDAR representation for each car. This helps us to alleviate the issue of far-away cars where intially their object point cloud is sparse and ambiguous, as well as remove potential false positives of the 2D detector as we discard any detections which cannot be tracked across at least three frames.

For the algorithm to work correctly, we need to make sure that we take into account vehicle movement - both the movement of the vehicle where the LiDAR&camera is mounted on (ego-vehicle), as well as movement of other vehicles around the ego-vehicle. In order to compensate the ego-motion of our vehicle, we use the available data from the Inertial measurement unit (IMU) unit. We however found out that the measurement are not always precise enough, and therefore we employ point to plane variant of the Iterative Closest Point (ICP) algorithm [22] with the IMU data as the prior estimate to improve LiDAR scans alignment.

In order to compensate for movement of vehicles around our car, we build a simple motion model to predict the location of each car based on the last known location and predicted velocity

$$\mathbf{p}_i^t = \mathbf{p}_i^{t-1} + (\mathbf{p}_i^{t-1} - \mathbf{p}_i^{t-2}) \tag{2}$$

where $\mathbf{p}_i^t$ denotes estimated 3D position $\mathbf{p}_i^t = (X_i^t,\, Y_i^t,\, Z_i^t)$ of the vehicle i in the frame t.

LiDAR Points Downsampling. For the method to stay computationally feasible (note that in Eq. (1) we sum over all points in L_i) we limit the number of aggregated points by combining two different downsampling strategies: we use voxel downsampling to reduce redundant points and to increase the possibility of seeing more parts of the car, and we combine it with random downsampling, which reduces outliers but retains points seen many times. The object point cloud L_i is created by concatenating results of both downsampling strategies.

3.2 Appereance Mask Loss (AML)

The second loss function is the *Appereance Mask Loss* which ensures predicted object locations in 3D are consistent with 2D observations from the RGB camera (see Figure 4). This is especially useful for situations which are ambiguous in the LiDAR modality alone, like object occlusions or the vertical boundaries of a vehicle, because for example often it's not obvious from the LiDAR which points correspond to the vehicle and which to the road.

The Apperance Mask loss is based on the per-pixel Binary Cross Entropy (BCE) which compares the observed object mask $B_i \in (0,1)^{W \times H}$ from the 2D instance segmentation [31] to the mask of the corresponding network prediction through differentiable renderer $\mathcal{R}$

$$\mathcal{L}_{\text{AML}}(B_i, P_i, M) = \frac{1}{|B_i|}\text{BCE}\left(B_i, \mathcal{R}(M, P_i)\right) \tag{3}$$

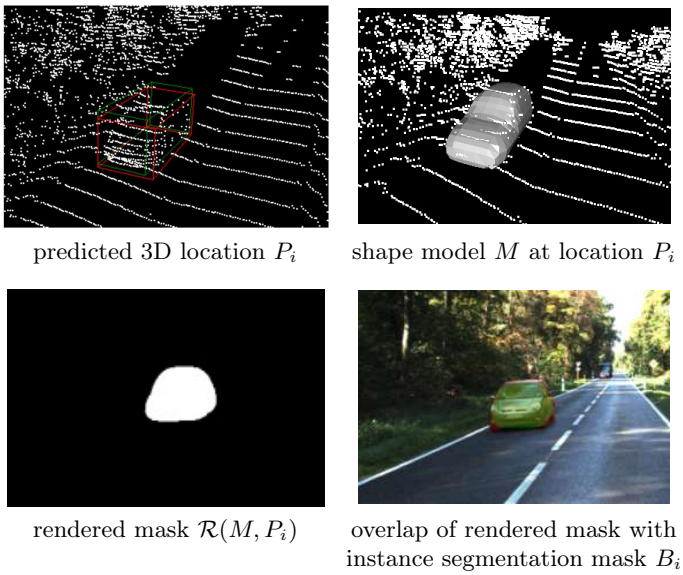

Fig. 4: Appearance Mask Loss (AML) ensures predicted object location in the 3D world is consistent with object observation in the RGB camera (bottom right figure: rendered mask in green, instance segmentation from RGB camera B_i in red).

where $\mathcal{R}(M, P_i)$ denotes rendered mask of the shape model M placed in the position P_i in the 3D scene.

3.3 Multiple shape hypotheses

It is important to note that cars come in few basic different shapes, such as a hatchback or sedan, and trying to fit a hatchback template to a sedan would lead to sub-optimal results. In our method, we therefore define a set of four basic car shapes $\mathcal{M} = \{M_{\text{hatchback}}, M_{\text{sedan}}, M_{\text{SUV}}, M_{\text{MPV}}\}$ and our final loss only considers the template where the individual loss is minimal. This allows the network to maintain multiple hypothesis during the training, as it is not forced to make hard decisions early on.

$$\mathcal{L}(L, P) = \sum_{i=1}^{N} \arg\min_{M \in \mathcal{M}} \Big(\mathcal{L}_{\text{TFL}}(L_i, P_i, M) + \lambda \, \mathcal{L}_{\text{AML}}(B_i, P_i, M) \Big) \tag{4}$$

3.4 Training process

We use Voxel-RCNN [2] architecture – an architecture introduced for fully-supervised 3D object detector – and add loss functions with our loss formulation of Equation (4). We also observed that training convergence can be significantly

sped up if we keep the original losses of Voxel-RCNN, by replacing human labels with crude pseudo ground truth. Before starting the training process, we therefore generate pseudo ground truth by iteratively enumerating all possible values for $(X_i, Y_i, Z_i, \theta_i, H_i, S_i)$ in a given range, and select the configuration with the lowest Template Fitting Loss (see Eq. (1)) as an initial estimate of vehicle position. This offline process therefore creates very crude pseudo ground truth, but our full loss Eq. (4) formulation allows the network to learn more fine-grained predictions and achieve better accuracy.

4 Experiments

4.1 Dataset

We primarily conduct experiments on the standard KITTI dataset [5] on the Car category, consistently with all previous weakly-supervised methods [11, 12, 14, 15, 19]. We additionally evaluate our method on Waymo Open Perception Dataset [26] (WOD), as the first weakly-supervised method to our knowledge.

Table 1: 3D object (car) detection Average Precision on KITTI validation set.

		Human annotations		BEV AP						3D AP					
				Easy		Moderate		Hard		Easy		Moderate		Hard	
Method	Year	2D	3D	@0.5	@0.7	@0.5	@0.7	@0.5	@0.7	@0.5	@0.7	@0.5	@0.7	@0.5	@0.7
Fully-supervised methods															
PV-RCNN [23]	2020	yes	yes	✗	✗	✗	✗	✗	✗	✗	89.4	✗	83.7	✗	78.7
Voxel-RCNN [2]	2021	yes	yes	✗	✗	✗	✗	✗	✗	✗	89.4	✗	84.5	✗	78.9
CasA+T [30]	2021	yes	yes	✗	✗	✗	✗	✗	✗	✗	**90.1**	✗	**86.6**	✗	**79.5**
Weakly-supervised methods with partial (500 frames) human labels															
WS3D [16]	2020	no	partial	96.3	88.6	89.0	85.0	88.5	84.7	95.9	84.0	89.1	75.1	88.3	73.3
FGR [29]	2021	yes	no	✗	✗	✗	✗	✗	✗	✗	86.1	✗	74.9	✗	67.5
WS3D v2 [15]	2021	no	partial	96.5	88.9	89.3	85.8	89.0	85.0	96.3	85.0	89.4	75.9	88.9	74.4
MAP-Gen [12]	2022	yes	partial	✗	✗	✗	✗	✗	✗	✗	87.9	✗	78.0	✗	76.1
Mtrans [11]	2022	yes	partial	✗	✗	✗	✗	✗	✗	✗	88.7	✗	78.8	✗	77.4
Weakly-supervised methods with no human labels															
VS3D [19]	2020	no	no	81.6	✗	72.4	✗	64.31	✗	41.8	✗	39.2	✗	32.7	✗
Zakharov [33]	2020	no	no	94.9	81.0	88.5	59.8	✗	✗	90.7	22.4	71.1	13.3	✗	✗
McCraith [14]	2022	no	no	90.2	✗	85.7	✗	76.8	✗	✗	✗	✗	✗	✗	✗
TCC-Det (ours)		no	no	**98.9**	**90.1**	**89.6**	**88.3**	**89.1**	**87.0**	**98.8**	85.9	**89.5**	75.3	**89.0**	73.7

The KITTI dataset contains 7481 training samples and 7518 testing samples. As the dataset does not contain a specific validation set, we use the same split as in [14, 29, 33], which has 3712 training samples and 3769 validation samples. Objects in the dataset are divided into three categories: Easy, Moderate and Hard, based on their occlusion, visibility and height of the bounding box in the camera.

The WOD contains approximately 150k training, 40k validation and 30k testing samples. The labelling process in 2D and 3D is not coupled. Therefore, many objects labelled in 3D are not captured in the camera's FOV. Vehicles fall into two categories (level 1 and level 2) based on the detection difficulty.

Object detection is evaluated using two metrics – 3D and Bird's Eye View (BEV) Average Precision. Both metrics are based on the standard intersection

over union (IoU) measurement. We provide results for both 0.5 and 0.7 IoU thresholds, to make sure we can compare with all previous methods, where some only report results for the 0.5 or the 0.7 threshold respectively.

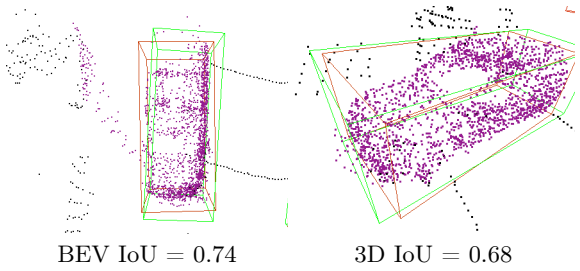

Fig. 5: An example on the KITTI dataset of a detection by our model (in red) which is considered correct in the BEV metric but incorrect in the 3D metric. We argue that given LiDAR point cloud of the vehicle (purple) our detection is correct, but methods relying on raw data like ours are unable to adjust to such annotation bias in its outputs. Detections in red, ground truth in green. Best viewed zoomed in.

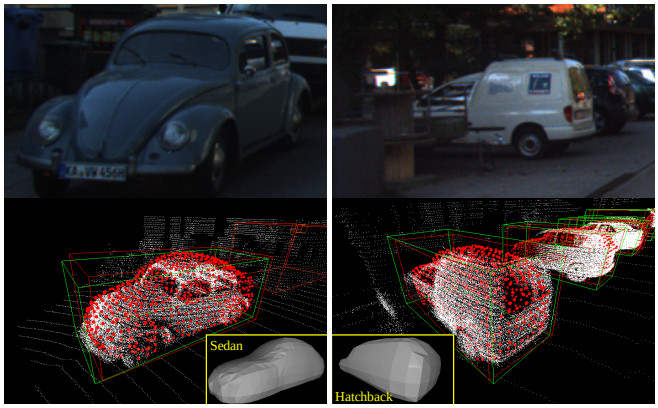

Fig. 6: Sample results on unusual cars. Template shape model M in position P_i with lowest loss value as red points / red bounding box, human annotation in green. The same shape model M shown as a mesh for illustrative purposes in the yellow cut out.

4.2 KITTI Implementation

The templates $M \in \mathcal{M}$ used for the Template Fitting Loss (see Eq. (1)) were created by uniformly sampling 1000 points from four 3D generic 3D models of

cars obtained online [1], where their size was initially adjusted to match the average car dimensions of the KITTI dataset. The templates $M \in \mathcal{M}$ generalize well to sufficiently fit rare shape cars (see Fig. 6).

Table 2: 3D object (car) detection Average Precision on KITTI test set. Results denoted as * were obtained reproducing published code.

		Human annotations		BEV AP			3D AP		
Method	Year	2D	3D	Easy @0.7	Moderate @0.7	Hard @0.7	Easy @0.7	Moderate @0.7	Hard @0.7
Fully-supervised methods									
PV-RCNN [23]	2020	yes	yes	95.1	90.7	86.1	90.3	81.4	76.8
Voxel-RCNN [2]	2021	yes	yes	94.9	88.8	86.1	90.9	81.6	77.1
CasA+T [30]	2021	yes	yes	**94.6**	**91.2**	**88.4**	**90.7**	**84.0**	**79.7**
Weakly-supervised methods with partial human labels									
WS3D [16]	2020	no	partial	90.1	84.0	77.0	80.1	69.6	63.7
FGR [29]	2021	yes	no	90.6	82.7	75.5	80.3	68.5	61.6
WS3D v2 [15]	2021	no	partial	91.0	84.9	78.0	81.0	70.6	64.2
MAP-Gen [12]	2022	yes	partial	90.6	85.9	80.6	81.5	74.1	67.6
Mtrans [11]	2022	yes	partial	91.4	86.0	78.8	83.4	75.1	68.3
Weakly-supervised methods with no human labels									
Zakharov * [33]	2020	no	no	78.5	72.3	64.5	36.6	26.0	21.7
TCC-Det (ours)		no	no	91.2	85.1	80.2	77.8	65.4	60.9

The steepness parameter k of the loss was set to 10. For the Appearance Mask Loss (see Eq. (3)) we used soft silhouette shader [13] for rendering of the template. To aggregate LiDAR detections, we used 30 frames before and 30 frames after the reference frame, which at the sample rate of 10 Hz is equal to 3 seconds before and 3 seconds after.

Detectron2 [31] with a pre-trained model on MS-COCO [10] dataset was used as the 2D object instance segmentation framework. Only detections with a score higher than 0.7 were considered.

The Voxel-RCNN [2] was pre-trained on the pseudo-ground truth for 50 epochs with batch size equal to 50 on 2x NVIDIA A100. The initial learning rate was 0.01 and the weight decay was 0.01. We used Adam optimizer [8] and Cosine Annealing learning rate scheduler with one epoch warmup. The weight of the Appearance Mask Loss λ was 0.1. It took approximately 2 hours to pretrain the network, while creating the pseudo ground truth labels took approximately 6 hours. The full training of the network with our loss function (Eq. (4)) was for 10 epochs with batch size 8 on 2x NVIDIA A100. The learning rate was set to 0.001 and weight decay 0.01. It took approximately 10 hours to train the network. We added our proposed loss functions only to the final stage (Detect Head) of the Voxel-RCNN [2] network, as it serves as a fine refinement, while the initial stage (RPN) loss was unchanged, only training on the pseudo ground truths. For WOD implementation, please refer to the supplementary material.

4.3 KITTI Results

On the KITTI validation set, our method *significantly outperforms all previous methods which do not rely on domain-specific human labels* [14,19,33] by a great margin, thus achieving state-of-the-art accuracy in 3D object detection trained without human annotations (see Table 1). Our method achieves similar 3D AP as FGR [29] in Easy and Moderate categories and outperforms it in the Hard category by a great margin, despite the fact FGR relies on 2D KITTI ground truth annotations in its training. Our method also outperforms WS3D [16] and WS3Dv2 [15] in BEV and achieve comparable results in the 3D AP, despite the fact both methods rely on 500 frames of human 3D KITTI labels for training. MAP-Gen [12] and Mtrans [11] achieve slightly better results than us in 3D AP, but they again, unlike our method, rely on 500 frames with 3D labels annotated by humans, and we are unfortunately unable to compare in the BEV metric because these results are not published for some reason.

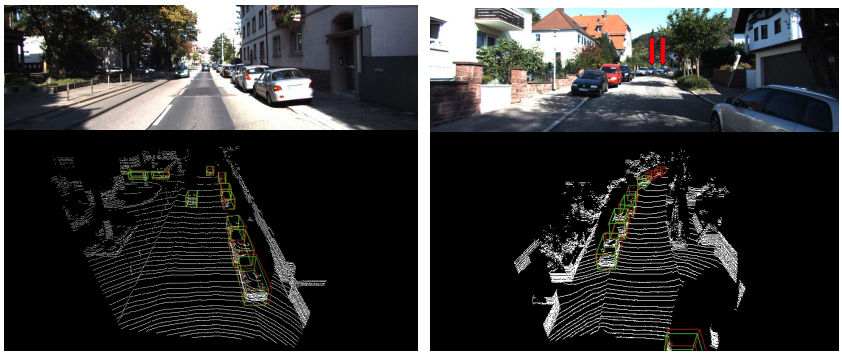

Fig. 7: Qualitative examples of 3D object detections on the KITTI dataset. Note the two detections marked with a red arrow which are cars missed by human annotators of KITTI. Detections in red, ground truth in green. Best viewed zoomed in.

On the KITTI test set, our method outperforms FGR [29], WS3D [16] and WS3Dv2 [15] using the BEV metric across all categories and achieves similar performance as MAP-Gen [12] and Mtrans [11] (see Table 2). As none of the previous methods that do not use human labels [14, 19, 33] reports results on the KITTI test set, we are only able to compare to Zakharov et al. [33] as test labels can be easily obtained from their published code. Using the 3D metric, a small performance gap is still present when compared to methods which use partial 3D human labels, which we believe is due to an annotation bias in KITTI (see Figure 5) which makes it very hard to reach the required IoU of 0.7 without somehow incorporating this bias into our method (we speculate that the methods which use partial 3D human labels actually are able to learn this bias as part of their training process).

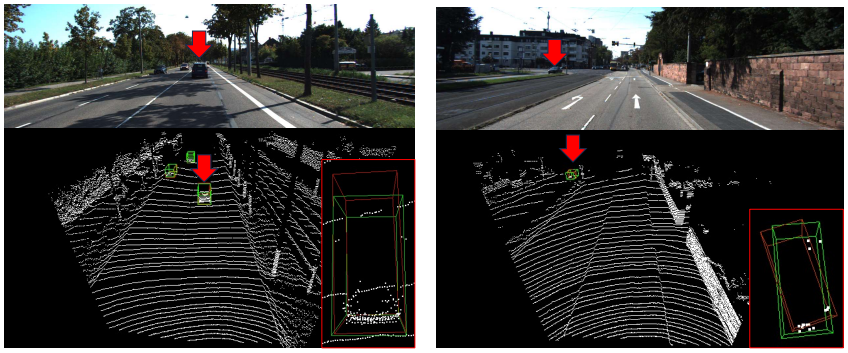

Fig. 8: Typical failure modes on the KITTI dataset. Estimating length of a car which is moving and has the same yaw as the ego-vehicle is extremely difficult as there is no data even in subsequent frames to infer vehicle length (left). A vehicle in the Hard category (car) has very sparse LiDAR point cloud and since it is a moving car in opposite direction, we cannot aggregate enough LiDAR points for this instance (right).

Table 3: 3D object (car) detection Average Precision on the Waymo Open Perception [26] validation set.

		BEV AP				3D AP			
	Human	Level 1		Level 2		Level 1		Level 2	
Method	labels	@0.5	@0.7	@0.5	@0.7	@0.5	@0.7	@0.5	@0.7
Voxel-RCNN [2]	3D boxes	95.80	91.00	91.47	84.40	94.17	77.37	88.58	69.04
TCC-Det (ours)	None	76.81	58.27	68.97	51.36	69.98	17.44	62.24	15.01

4.4 Waymo Open Perception Dataset Results

Despite using no 3D human labels, our method achieves competitive results on the WOD (see Tab. 3 and Fig. 9), especially in the Birds Eye View (BEV) metric. The main limitation is that in this dataset cameras do not cover the whole area covered by LiDAR, and therefore some vehicles are inherently not included in our training: our method can actually use stationary cars out of cameras' FOV in the training, because thanks to our proposed temporal consistency exploitation such car is at some point captured by the camera as the ego-vehicle drives around; on the other hand, cars moving behind the ego-vehicle are often only captured in LiDAR and never by a camera, and as such are excluded. Note that limitation is consistent with previous methods such as Mtrans [11] and MAP-Gen [12] and this limitation can be solved by including more cameras into the setup, to make sure cameras' and LiDAR FOV sufficiently overlap.

4.5 Ablations

In this section, we present three ablation studies to support the design choices we made, additional ablations, such as the number of frames, steepness parameter k and ICP, are presented in Supplementary material. All ablations are evaluated on the KITTI dataset.

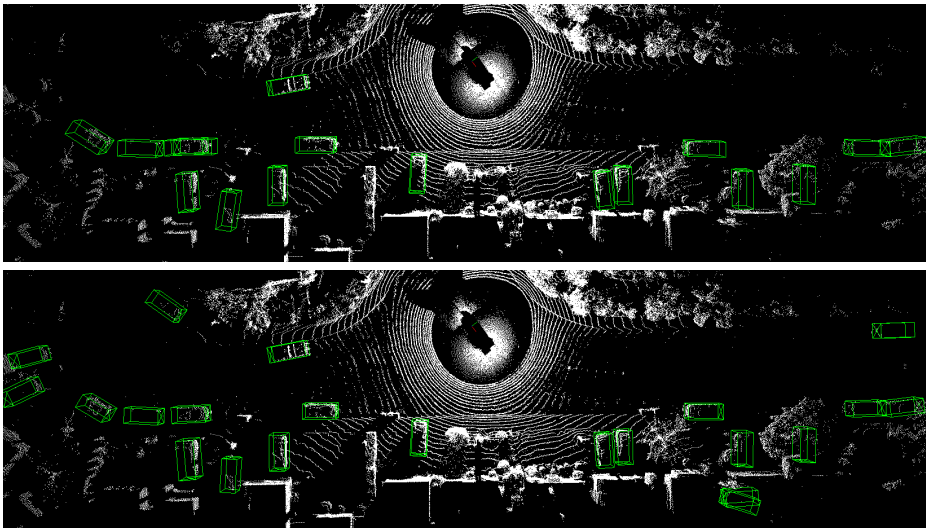

Fig. 9: Qualitative examples on the Waymo Open Perception dataset [26]. Our method (top) and fully-supervised Voxel-RCNN [2] (bottom).

Individual loss functions. In our paper, we introduce two new loss functions – Template Fitting Loss (TFL) and Appearance Mask Loss (AML). In Table 4, we show the impact of using just one of the losses. Template Fitting Loss (Eq. (1)) on its own shows a small improvement in BEV and 3D, Appearance Mask Loss (Eq. (3)) on its own greatly improves 3D Average Precision, but combining both losses brings the biggest benefit.

Table 4: Ablation study of the proposed loss functions on the KITTI validation set.

Loss		BEV			3D		
		Easy	Moderate	Hard	Easy	Moderate	Hard
TFL	AML	@0.7	@0.7	@0.7	@0.7	@0.7	@0.7
✗	✗	89.64	87.55	85.48	78.42	64.77	63.64
✓	✗	90.12	88.15	86.99	76.42	66.01	65.00
✗	✓	90.09	88.05	86.64	85.35	74.54	67.67
✓	✓	90.09	88.25	86.95	85.92	75.33	73.74

Chamfer distance loss. Another ablation study to support using Template Fitting Loss instead of the well-known Chamfer distance loss or McCraith et al. loss [14] is shown in Table 5, where the Template Fitting Loss greatly improves accuracy over the Chamfer distance loss and McCraith et al. loss in both BEV and 3D Average Precision. The Appearance Mask Loss was present in all three cases.

Training dataset size. The main advantage of weakly-supervised methods is the ability to exploit larger volumes of training data, because the training data do not require human labels. We unfortunately do not have more unlabelled data available from the KITTI dataset, but instead to demonstrate this ability of our

Table 5: Ablation study of the Template Fitting loss function on the KITTI validation. * denotes reproducible loss without discounting outliers from McCraith et al. [14].

Point cloud distance	BEV			3D		
	Easy @0.7	Moderate @0.7	Hard @0.7	Easy @0.7	Moderate @0.7	Hard @0.7
McCraith loss [14] *	77.30	75.37	66.41	53.45	49.76	42.80
Chamfer loss	84.02	73.11	66.39	67.21	53.93	51.37
TFL	90.09	88.25	86.95	85.92	75.33	73.74

method, we assume that only part of the dataset had actually been labelled while the rest remained unlabelled. As it is demonstrated in Tab. 6, for example when using 25% of human labels to fine-tune our trained model, our method performs almost identically to the fully-supervised model – in another words, the time and money originally spent on creating 75% of KITTI labels could have been saved.

Table 6: Ablation study of the impact of the amount of human labels

Fraction of human labels	BEV			3D		
	Easy @0.7	Moderate @0.7	Hard @0.7	Easy @0.7	Moderate @0.7	Hard @0.7
0 % (Ours)	90.09	88.25	86.95	85.92	75.33	73.74
10 %	90.27	87.78	86.10	88.95	78.58	77.42
25 %	90.25	87.78	87.05	89.27	78.98	78.13
50 %	90.22	88.26	87.64	89.41	79.24	78.51
100 %	90.44	88.39	87.86	89.42	84.06	78.77

5 Conclusion

A new method which exploits a generic off-the-shelf 2D detector and a number of real-world priors to train a 3D object detector was proposed. The method can be used to train a 3D detector by only collecting sensor recordings in the real world, which is extremely cheap and allows training using orders of magnitude more data than traditional fully-supervised methods.

Our method *significantly outperforms all previous methods which do not rely on domain-specific human labels*, thus achieving state-of-the-art accuracy in 3D object detection trained without human annotations. In Bird's Eye View (BEV) AP, our method also outperforms methods which rely on partial 2D or 3D KITTI annotations, and in 3D AP it achieves similar results, despite not having access to any 3D human labels. We also show in ablations, that using only 25% of human labels, we get almost identical accuracy to the fully-supervised method.

The main limitation seems to be the inability to account for some annotation bias, which is demonstrated by a smaller gap to the fully-supervised method in the less strict overlap evaluation (IoU threshold 0.5).

Acknowledgement

The research was supported by Czech Science Foundation Grant No. 24-10738M. The access to the computational infrastructure of the OP VVV funded project CZ.02.1.01/0.0/0.0/16_019/0000765 "Research Center for Informatics" and of the Ministry of Education, Youth and Sports of the Czech Republic project e-INFRA CZ (ID:90254) is also gratefully acknowledged.

References

1. Free3d.com: 3d models for free. https://free3d.com/, accessed: 2023-11-14
2. Deng, J., Shi, S., Li, P., Zhou, W., Zhang, Y., Li, H.: Voxel r-cnn: Towards high performance voxel-based 3d object detection. vol. 35, pp. 1201–1209 (May 2021). https://doi.org/10.1609/aaai.v35i2.16207, https://ojs.aaai.org/index.php/AAAI/article/view/16207
3. Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM **24**(6), 381–395 (1981)
4. Gaidon, A., Wang, Q., Cabon, Y., Vig, E.: Virtual worlds as proxy for multi-object tracking analysis. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4340–4349 (2016)
5. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the kitti vision benchmark suite. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2012)
6. Hu, A., Russell, L., Yeo, H., Murez, Z., Fedoseev, G., Kendall, A., Shotton, J., Corrado, G.: Gaia-1: A generative world model for autonomous driving (2023)
7. Huang, X., Wang, P., Cheng, X., Zhou, D., Geng, Q., Yang, R.: The apolloscape open dataset for autonomous driving and its application. IEEE transactions on pattern analysis and machine intelligence **42**(10), 2702–2719 (2019)
8. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
9. Lang, A.H., Vora, S., Caesar, H., Zhou, L., Yang, J., Beijbom, O.: Pointpillars: Fast encoders for object detection from point clouds. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 12697–12705 (2019)
10. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: Common objects in context. In: ECCV. pp. 740–755. Springer (2014)
11. Liu, C., Qian, X., Huang, B., Qi, X., Lam, E., Tan, S.C., Wong, N.: Multimodal transformer for automatic 3d annotation and object detection. In: Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T. (eds.) Computer Vision – ECCV 2022. pp. 657–673. Springer Nature Switzerland, Cham (2022)
12. Liu, C., Qian, X., Qi, X., Lam, E.Y., Tan, S.C., Wong, N.: MAP-Gen: An automated 3D-box annotation flow with multimodal attention point generator. In: ICPR. pp. 1148–1155 (2022)
13. Liu, S., Li, T., Chen, W., Li, H.: Soft rasterizer: A differentiable renderer for image-based 3d reasoning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 7708–7717 (2019)

14. McCraith, R., Insafutdinov, E., Neumann, L., Vedaldi, A.: Lifting 2d object locations to 3d by discounting LIDAR outliers across objects and views. In: 2022 International Conference on Robotics and Automation (ICRA). pp. 2411–2418 (2022)
15. Meng, Q., Wang, W., Zhou, T., Shen, J., Jia, Y., Van Gool, L.: Towards a weakly supervised framework for 3d point cloud object detection and annotation. vol. 44, pp. 4454–4468 (2022). https://doi.org/10.1109/TPAMI.2021.3063611
16. Meng, Q., Wang, W., Zhou, T., Shen, J., Van Gool, L., Dai, D.: Weakly supervised 3d object detection from lidar point cloud. In: ECCV. pp. 515–531. Springer (2020)
17. Park, J.J., Florence, P., Straub, J., Newcombe, R., Lovegrove, S.: Deepsdf: Learning continuous signed distance functions for shape representation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 165–174 (2019)
18. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 652–660 (2017)
19. Qin, Z., Wang, J., Lu, Y.: Weakly supervised 3d object detection from point clouds. vol. abs/2007.13970 (2020), https://arxiv.org/abs/2007.13970
20. Richter, S.R., Vineet, V., Roth, S., Koltun, V.: Playing for data: Ground truth from computer games. In: Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14. pp. 102–118. Springer (2016)
21. Ros, G., Sellart, L., Materzynska, J., Vazquez, D., Lopez, A.M.: The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3234–3243 (2016)
22. Rusinkiewicz, S., Levoy, M.: Efficient variants of the icp algorithm. In: Proceedings Third International Conference on 3-D Digital Imaging and Modeling. pp. 145–152 (2001). https://doi.org/10.1109/IM.2001.924423
23. Shi, S., Guo, C., Jiang, L., Wang, Z., Shi, J., Wang, X., Li, H.: Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 10529–10538 (2020)
24. Shi, S., Wang, X., Li, H.: Pointrcnn: 3d object proposal generation and detection from point cloud. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 770–779 (2019)
25. Song, S., Lichtenberg, S.P., Xiao, J.: Sun rgb-d: A rgb-d scene understanding benchmark suite. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 567–576 (2015)
26. Sun, P., Kretzschmar, H., Dotiwalla, X., Chouard, A., Patnaik, V., Tsui, P., Guo, J., Zhou, Y., Chai, Y., Caine, B., et al.: Scalability in perception for autonomous driving: Waymo open dataset. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 2446–2454 (2020)
27. Sun, T., Segu, M., Postels, J., Wang, Y., Van Gool, L., Schiele, B., Tombari, F., Yu, F.: SHIFT: a synthetic driving dataset for continuous multi-task domain adaptation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 21371–21382 (2022)
28. Wang, Y., Chen, X., You, Y., Li, L.E., Hariharan, B., Campbell, M., Weinberger, K.Q., Chao, W.L.: Train in germany, test in the usa: Making 3d object detectors generalize. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11713–11723 (2020)

29. Wei, Y., Su, S., Lu, J., Zhou, J.: Fgr: Frustum-aware geometric reasoning for weakly supervised 3d vehicle detection. In: 2021 IEEE International Conference on Robotics and Automation (ICRA). pp. 4348–4354. IEEE (2021)
30. Wu, H., Deng, J., Wen, C., Li, X., Wang, C., Li, J.: Casa: A cascade attention network for 3-d object detection from lidar point clouds. IEEE Transactions on Geoscience and Remote Sensing **60**, 1–11 (2022)
31. Wu, Y., Kirillov, A., Massa, F., Lo, W.Y., Girshick, R.: Detectron2. `https://github.com/facebookresearch/detectron2` (2019)
32. Xu, R., Xiang, H., Xia, X., Han, X., Li, J., Ma, J.: OPV2V: An open benchmark dataset and fusion pipeline for perception with vehicle-to-vehicle communication. In: 2022 International Conference on Robotics and Automation (ICRA). pp. 2583–2589. IEEE (2022)
33. Zakharov, S., Kehl, W., Bhargava, A., Gaidon, A.: Autolabeling 3d objects with differentiable rendering of SDF shape priors. In: CVPR. pp. 12224–12233 (2020)