2.5D-VoteNet: Depth Map based 3D Object Detection for Real-Time Applications

Lanxiao Li
lanxiao.li@kit.edu
Michael Heizmann
michael.heizmann@kit.edu

Institute of Industrial Information Technology
Karlsruhe Institute of Technology
Karlsruhe, Germany

Abstract

We address the 3D object detection task by capturing features directly on depth maps with a 2D CNN. Most existing 3D object detection methods take point clouds as input, even when each point cloud is converted from a single depth map. Although they have achieved impressive performance, point cloud based 3D detectors usually have high computational cost and complex structure, which limits their application on mobile devices and in real-time scenarios. Building on the state-of-the-art VoteNet [29], we propose 2.5D-VoteNet, a powerful and efficient depth map based 3D detection pipeline. Since our models extract features directly on depth maps, most computation remains in 2D space and can be efficiently executed. Instead of using an off-the-shelf 2D CNN, we introduce relative depth convolution (RDConv) to learn robust local features. Our end-to-end pipeline achieves state-of-the-art results on the challenging SUN RGB-D [39] benchmark and surpasses the baseline with a clear margin on ScanNet [8] frame-level detection task. Meanwhile, our method reaches a significantly higher inference speed than existing methods (69 FPS).

1 Introduction

Many methods have been proposed for 3D object detection on point cloud data. Roughly speaking, they can be separated into projection-based methods [5, 6, 17, 19, 20, 35, 36, 49, 54] and point cloud based methods [3, 7, 11, 13, 16, 29, 33, 34, 40]. Projection-based methods project point clouds into bird eye view, front view or multiple views. By doing this, point clouds are converted into images, so that mature 2D CNNs can be applied. However, they are usually unsuitable for indoor scenarios because objects are more cluttered than outdoors. Point cloud based methods use raw or quantized (voxelized) point clouds as input. In order to aggregate local and global features from point clouds, they often hierarchically downsample the input and perform convolution-like operations. Since point clouds are sparse and irregular, these operations are computationally expensive due to irregular memory access and dynamic kernel overhead [25]. Although they have achieved promising performance, point cloud based methods are unable to achieve high inference speed. The state-of-the-art methods for indoor scenes have typically a frame rate of $\sim$10 FPS [3, 11, 29, 30, 44, 50]

and cannot fulfil the requirement in some real-time applications. Moreover, the high computational cost and special self-defined operators in point cloud based methods (*e.g.* PointNet++ [31], sparse 3D convolution in [11]) prohibit hardware implementations on a lot of mobile or small devices (*e.g.* cell phones, small robots, drones), since they support solely efficient processors with either less computational resources (*e.g.* ARM CPU, mobile GPU) or limited operator support (*e.g.* neural ASIC, NPU).

Our work is motivated by the observation that, in many applications, especially in indoor scenarios, a point cloud is converted from a single depth map. Given the camera calibration, depth maps and point clouds are two different representations of exactly the same information. Thus, it should be possible to perform 3D object detection directly on depth maps instead of point clouds. In this work, we capture features on depth maps, while keeping the detection head and post-processing in 3D space. Unlike existing 2D CNNs, which use absolute depth as input, the backbone of our network is augmented with novel relative depth convolution (RDConv), which learns local features from relative depth information. The RDConv is motivated by the simple intuition that local geometries (*e.g.* edges, corners) depend more on relative depth and are invariant to the absolute depth. Compared to the common convolution using absolute depth, RDConv extracts more informative features and improves the detection result. Our pipeline shows higher inference speed and better hardware-friendliness than point cloud based methods, while reaching competitive performance.

Our contributions in this work are as follows: (1) We introduce a simple and efficient depth map based 3D detection pipeline. (2) We propose the novel approach of relative depth convolution (RDConv), which is effective at capturing local information on depth maps. (3) Our models achieve the state-of-the-art results on the challenging SUN RGB-D [39] benchmark and are significantly faster than existing methods. (4) Our models outperform the baseline VoteNet [29] in the ScanNet [8] frame-level detection task with a clear margin.

2 Related Works

3D Detection on Point Clouds. Bird eye view based methods [5, 6, 17, 19, 35, 36, 49, 54] project LiDAR point clouds to the ground plane to generate pseudo-images and are widely used for autonomous driving. However, they are unsuitable in cluttered indoor scenes. For 3D detection in indoor scenarios, [18] estimates object orientations via Manhattan Frame Estimation (MFE) and performs bounding box regression by feeding 3D coordinates histogram to a multi-layer perceptron. Deep Sliding Shape [37] uses Truncated Signed Distance Function (TSDF) to encode depth maps and generates 3D proposal via dense 3D convolution. Although these methods use RGB-D images as input, they don't directly aggregate features on 2D images. VoteNet [29] proposes an end-to-end pipeline for 3D detection in point cloud and uses deep Hough voting to centralize the point features. Recent works [3, 30, 44, 45, 50] further optimize VoteNet to improve the prediction quality. However, most variants don't take the inference speed into consideration. Based on sparse 3D convolution, GSDN [11] proposes generative methods to compensate the sparsity of point clouds. But this approach is only applicable on 3D scans.

RGB Fusion in 3D Detection. [13, 28, 48] generate 2D regions of interest (RoI) on RGB images using a pre-trained 2D detector. Objects within each 2D RoI are then detected via point cloud based networks. [13, 14, 38] enrich point features with high-level color features from RGB images. ImVoteNet [30] uses a 2D detector to gain extra image votes. Due to the late fusion strategy, the computational cost of these methods is inevitably high.

2D CNN for Range Images and Depth Maps. Unlike the bird eye view based methods, some works [5, 20, 27] project LiDAR point clouds to the front view. However, they directly predict 3D bounding boxes on 2D feature maps and consequently lose precise 3D spatial information. RangeRCNN [23] combines the LiDAR range image with point view and bird eye view representation for 3D detection. However, the pipeline is still computationally expensive due to the complicate structure. Bewley *et al*. [2] propose range conditioned dilated convolution for scale invariant 3D detection on range images, while our method addresses the absolute depth invariance of local features. Besides the detection tasks, 2D CNNs and RGB-D images have been widely utilized in semantic segmentation and salient object detection tasks. A lot of works focus on suitable structures for fusing features from two modalities [4, 15, 21, 22, 26, 51, 52]. Some works [43, 46, 47] mimic the behavior of 3D CNNs with 2D CNNs. However, our work shows that capturing geometrical features on depth maps doesn't necessarily mean to mimic 3D CNNs.

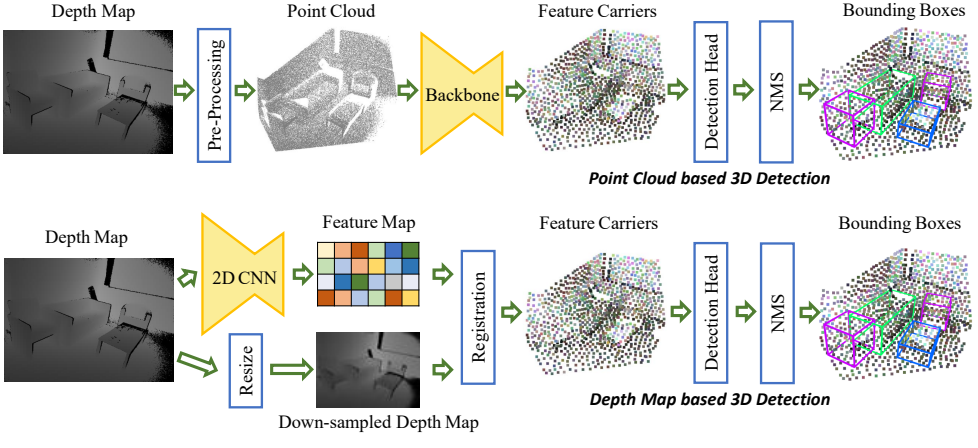

Figure 1: Point cloud based and depth map based 3D detection pipelines. We take the image sample from the SUN RGB-D dataset [39].

3 Depth Map based 3D Detection

In this section we first briefly revisit the pipeline of point cloud based 3D detection (Sec. 3.1). Later, we introduce our depth map based pipeline (Sec. 3.2). Then, we introduce an important ingredient of our method: relative depth convolution (RDConv) (Sec. 3.3).

3.1 Point Cloud based Pipeline

The common workflow of a point cloud based 3D object detection can be separated into four components: the pre-processing, a 3D backbone, a detection head and the post-processing. As illustrated in Fig. 1 , depth maps are converted into point clouds at pre-processing stage. Usually, point clouds are further down-sampled or quantized to reduce the computational cost. The backbone takes sparse point coordinates to conduct convolution-like operations within local spherical neighborhood [29, 31], on sparse grids [7, 11] or within local regions

defined by K-nearest-neighbors [5, 34]. The output of the backbone is a down-sampled subset of the input point cloud with high level features attached to each point (also called feature carriers or points of interest). The detection head then aggregates object-relevant information and performs bounding box regression and semantic classification. In post-precessing stage, non-maximum suppression (NMS) is applied to remove redundant predictions.

3.2 Depth Map based Pipeline

Our pipeline is shown in Fig. 1 . We use a U-shaped 2D CNN to encode the high-resolution depth map into a low-resolution feature map. Thus, the feature aggregation takes place in 2D space rather than in 3D space. However, to fully utilize the spatial information, we keep feature carriers, the detection head and the NMS in 3D space. To build feature carriers, the input depth map is scaled to the same resolution as the feature map and lifted into 3D space. Then, each feature vector in the feature map is registered to the corresponding 3D point. Since feature carriers are in 3D space and can be processed as in the point cloud based pipeline, we adopt the detection head (with the voting module) and the post-processing from VoteNet [29]. Thus, we name our model 2.5D-VoteNet.

Compared to point cloud based methods, our depth map based pipeline brings multiple advantages. First, the network is accelerated thanks to the efficiency of the 2D CNN. Moreover, the low computational cost allows a deeper backbone and higher-resolution input, which results in more informative features and better detection quality. Point cloud based networks, however, have usually limited model depth and only accept down-sampled or voxelized point clouds to avoid excessive run-time and memory usage. Also, our simple design simplifies the hardware implementation in real-world applications. Meanwhile, our pipeline simplifies the fusion of geometrical and color information, since both the depth map and color map have 2D grid structures and can be scaled to the same resolution. On the contrary, it's nontrivial to incorporate RGB images into the point cloud based pipeline due to different data properties (*e.g.* regularity vs. irregularity, high resolution vs. low resolution), as discussed in [13, 14, 30]. Specifically, we gain a significant performance boost by adding only one layer to accept RGB images. The fusion strategy is further explained in Sec. 4.

The main concern of the depth map based 3D detection is that the 2D backbone is unaware of the camera calibration. However, our experiments imply that our models learn calibration invariance via data augmentation (see supplementary material). The limitation of our method is that it cannot directly handle 3D scans, which are reconstructed from multiple views of depth maps.

3.3 Relative Depth Convolution

In this work, we propose a novel convolution operation which depends on relative depth rather than absolute depth. The intuition behind this design is trivial: local geometries (*e.g.* edges and corners) rely more on relative depth than absolute depth. The observation implies that absolute depth invariance might help a 2D CNN to capture more informative and robust features on depth maps.

Following this intuition, the proposed relative depth convolution (RDConv) normalizes the depth values in each sliding window with respect to a local reference depth. The new operator negligibly increases the computational cost and can replace the first convolution layer in a common 2D CNN without changing its overall structure. Let $x \in \mathbb{Z}^2$ be an arbitrary

2D coordinate, $d(x): \mathbb{Z}^2 \mapsto \mathbb{R}$ a depth map and $y(x): \mathbb{Z}^2 \mapsto \mathbb{R}^n$ an n-dimensional feature map. The RDConv is defined as follows:

$$y(x) = \sum_{\Delta x \in \Omega} (d(x+\Delta x) - D_r(x)) \cdot M(x+\Delta x) \cdot w(\Delta x) \tag{1}$$

where $\Delta x \in \mathbb{Z}^2$ defines the offsets, $\Omega \subset \mathbb{Z}^2$ the set of offsets and $w(\Delta x): \mathbb{Z}^2 \mapsto \mathbb{R}^n$ learnable weights of kernels. $M(x)$ defines a binary mask with $M(x) = 0$ if $d(x) = 0$, and $M(x) = 1$ if $d(x) > 0$. $D_r(x)$ is the reference depth. In this work, we use the average depth within each sliding window as reference, which is defined with:

$$D_r(x) = \frac{\sum_{\Delta x \in \Omega} M(x+\Delta x) \cdot d(x+\Delta x)}{\sum_{\Delta x \in \Omega} M(x+\Delta x) + \varepsilon} \tag{2}$$

Here ε is a small constant to avoid dividing by zero. The binary mask $M(x)$ is necessary since depth maps might contain bad pixels (*e.g.* dark pixels on the depth map in Fig. 1). It means their depth values are not measurable or invalid. The depth of bad pixels is usually set to zero by the pre-processing. The binary mask $M(x)$ omits bad pixels for the convolution kernel and for the relative depth calculation.

Discussions on masking. Since the depth values of bad pixels are zeroed by pre-processing, normal 2D Conv as the first layer already benefits from the masking effect. The advantage of RDConv comes therefore from the local depth normalization rather than the binary mask. Unlike sparsity invariant convolution [41], we don't apply binary masks in deeper layers. First, [41] focuses on very sparse 2D data, while depth maps in indoor scenes have limited sparsity. Also, the missing information at bad pixels can be recovered in deeper layers, as pixels share information with neighbors during the forward propagation. We empirically found that masking at deep layers brings no benefits to our pipeline.

4 Implementation Details

In this section, we introduce the detailed architecture and configuration of 2.5D-VoteNet.

Backbone and Fusion. We build up our backbone based on ResNet-34 [12]. As shown in Fig. 2, we modify ResNet-34 by adding an RDConv layer parallel to the first Conv layer. We don't remove the first Conv layer, in order to preserve information in absolute depth. We add the outputs of RDConv and common Conv for simplicity, as we empirically found adding and concatenation generate similar results. However, experiments show that our network also works without absolute depth. Unlike [12], we down-sample the feature map after the first Conv layer with strided convolution instead of pooling. Also, we reduce the output channels of the last Conv-Block in ResNet-34 from 512 to 256. We add skip connections between down-sampling and up-sampling parts. The output of the backbone is a feature map down-sampled by 8 and with 256 channels.

Until now, our network utilizes solely the geometrical information (depth map). For RGB fusion, we add an extra 2D Conv layer to accept RGB images at the beginning of the network. Then, the RGB features and the geometrical features are concatenated. We intend to use this early fusion strategy for efficiency and simplicity. In the fused model, the channel numbers of all 7×7 layers are reduced from 64 to 32, so that the first scale level has the same output channel number as in the geometry-only model.

Feature Carriers. To build the feature carriers, the depth map is resized to the same resolution as the output feature map from the backbone. With camera calibrations, the scaled

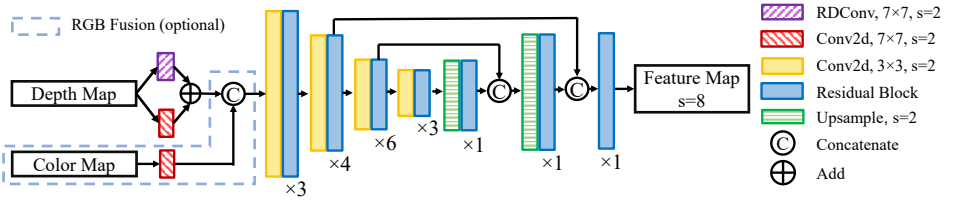

Figure 2: Architecture of our 2D backbone. Up-sample layers are followed by LeakyReLU and batch normalization (BN). RDCov and other Conv layers are followed by ReLU and BN. We use 2×2 transposed convolution to up-sample the feature maps.

depth map is lifted to the up-right coordinate system. We further sample 1024 points using farthest point sampling.

Detection Head. We adopt the detection head, consisting of a voting module and a proposal module, and the loss function from VoteNet [29]. The voting module predicts object centers and centralizes the point features. The proposal module clusters the point features and generates bounding box proposals.

Loss functions. Following [29], our network is trained end-to-end with a multi-task loss, defined as a weighted sum of a voting loss, an objectness loss, a 3D bounding box loss and a semantic classification loss. Furthermore, the box loss is composed of center regression, heading and size classification and GIoU sub-losses. One difference to [29] is that we omit the regression losses for heading angle residual and size residual and add a 3D-GIoU loss [53]. We refer readers to [29] and [53] for more details.

Input and Data Augmentation. The input depth maps and RGB images are scaled to 416×544. Zero padding is applied to keep the original aspect ratio. For data augmentation, we randomly resize the input by scaling the short edge by Uniform[320, 512]. Also, we randomly flip images and rotate them around their principal points by Uniform$[-15°, 15°]$. To improve the robustness against bad pixels, the depth values of 20% pixels are randomly set to zero. Also, the intrinsic and extrinsic parameters are augmented accordingly, when images are scaled, rotated and flipped.

Training on SUN RGB-D. The SUN RGB-D dataset [39] contains ∼10K RGB-D images of indoor scenes. We follow the official train/val split and use ∼5K scenes for training. We use Adam optimizer and train models for 160 epochs with initial learning rate 0.001 and batch size 16. The learning rate is decayed with factor 0.1 after 100 and 130 epochs, respectively.

Training on ScanNet. ScanNet [8] contains ∼1500 reconstructed 3D scans of indoor scenes with axis-aligned bounding box labels. In order to evaluate our models on ScanNet, we build a frame-level detection dataset based on it. Specifically, we unpack the raw data of ScanNet to gain ∼2M frames. We sample ∼100K RGB-D images for training and ∼10K for validation. The train/val split follows previous works [13, 29]. We project the scan-level bounding box labels to each camera coordinate system to gain the frame-level ground truth. In addition to our own models, we train a VoteNet on the frame-level ScanNet dataset as baseline. We train all models for 40 epochs and apply $\times 0.1$ learning rate decay after 20 and 30 epochs. Other configurations follow the SUN RGB-D training.

Inference. At inference time, we perform axis-aligned 3D NMS (C++ implementation) as post-processing. Following previous works, we validate our networks on the most common object classes and calculate the mean average precision (mAP) with 25% 3D-IoU threshold.

In this work, the inference speed (latency per frame) is measured on a PC with an Intel Core i7-8700 CPU and an Nvidia RTX 2080Ti GPU using Pytorch.

5 Experiments

In this section, we first compare 2.5D-VoteNet with SOTA methods (Sec. 5.1). Then, we show results of analysis studies and visualize qualitative results (Sec. 5.2). More experiments and visualizations can be found in supplementary materials.

5.1 Comparison with State-of-the-Art Methods

The state-of-the-art methods can be separated into methods without extra data and with extra data. The latter have components pretrained on other datasets (*e.g.* ImageNet [9], COCO [24], ScanNet [8]), whereas the former are directly trained on the target dataset.

Results on SUN RGB-D. We compare the geometry-only and fused version of 2.5D-VoteNet with SOTA methods on the SUN RGB-D benchmark. As shown in Tab. 1, both our models achieve competitive results and are significantly faster than previous methods. The geometry-only and fused model reach 3.1% and 4.0% higher mAP than the baseline VoteNet (57.7% mAP), respectively. Our fused model achieves the best results (61.7% mAP) among methods without extra data. Moreover, we explore the pretraining on ScanNet frame-level dataset. The pretraining further improves the mAP of our models by 1.2% and 2%, respectively. Our fused model then reaches the best mAP (63.7%) among all methods.

It's worth noting that several methods in comparison [3, 10, 30, 44, 45, 50] are optimized variants of VoteNet [29]. They aim to improve the mAP rather than the speed. Our work, however, significantly improves both aspects at the same time. As shown by the results, our models achieve impressive trade-off between detection accuracy and speed.

Results on ScanNet. The detection results on ScanNet frame-level detection task are shown in Tab. 2. We evaluate models with 18 object classes, following [13, 29]. Our geo-only and fused model achieves 6.8% and 7.2% higher mAP that the baseline VoteNet, respectively. The improvement on this dataset is more significant than on SUN RGB-D. We believe that it's because ScanNet frames are more challenging due to more partially visible objects and more hard samples, *e.g.* sinks, doors and pictures (see Fig. 5).

5.2 Analysis Experiments

In this section, we show results of analysis experiments. We use mAP with 25% 3D-IoU threshold on SUN RGB-D as metric of detection quality. We don't pretrain models on ScanNet in following experiments.

RDConv vs. Conv. Tab. 4 shows that, with common Conv as the first layer, the depth map based pipeline doesn't generate good detection result (56.1% mAP). Also, we found the training process unstable. It might be due to over-fitting, since common Conv takes absolute depth as input and learns relative spatial relations indirectly. By simply replacing the first layer with RDConv, the network gains a significant performance improvement ($\sim$ 4%) and the instability is also removed. The result proves that RDConv is the key component which enables our depth map based detection to learn informative features. While the network with one RDConv alone as the first layer already achieves competitive performance (60.4% mAP),

Methods	Input	ED	batht.	bed	booksh.	chair	desk	dresser	nightst.	sofa	table	toilet	Time	mAP
DSS [33]	P+I	✓	44.2	78.8	11.9	61.2	20.5	6.4	15.4	53.5	50.3	78.9	19.55*s*	42.1
PointFusion [43]	P+I	✓	37.3	68.6	37.7	55.1	17.2	23.9	32.3	53.8	31.0	83.8	1.3*s*	45.4
F-PointNet [28]	P+I	✓	43.3	81.1	33.3	64.2	24.7	32.0	58.1	61.1	51.1	90.9	0.12*s*	54.0
PointContrast [45]	V	✓	-	-	-	-	-	-	-	-	-	-	-	57.5
ImVoteNet [30]	P+I	✓	75.9	87.6	41.3	**76.7**	28.7	41.4	**69.9**	**70.7**	51.1	90.5	-	63.4
**Ours* (geo-only)	D	✓	75.3	88.1	41.6	75.6	29.2	39.1	61.0	70.5	49.7	89.8	**14.5***ms*	62.0
**Ours* (fused)	D+I	✓	76.4	87.1	**50.5**	75.0	31.2	41.4	64.4	**70.7**	49.7	90.3	**14.5***ms*	**63.7**
COG [32]	P+I	x	58.3	63.7	31.8	62.2	**45.2**	15.5	27.4	51.0	51.3	70.1	10*min*	47.6
Density based [1]	P	x	**79.4**	**88.2**	32.1	17.0	37.4	**53.7**	50.0	65.3	**53.3**	**95.8**	1.24*s*	57.2
VoteNet [29]	P	x	74.4	83.0	28.8	75.3	22.0	29.8	62.2	64.0	47.3	90.1	0.1*s*	57.7
MLCVNet [42]	P	x	79.2	85.8	31.9	75.8	26.5	31.3	61.5	66.3	50.4	89.1	-	59.8
EPNet [12]	P+I	x	75.4	85.2	35.4	75.0	26.1	31.3	62.0	67.2	52.1	88.2	-	59.8
H3DNet [50]	P	x	-	-	-	-	-	-	-	-	-	-	0.15*s*	60.1
SPOT [10]	P	x	-	-	-	-	-	-	-	-	-	-	-	60.4
HGNet [6]	P	x	78.0	84.5	35.7	75.2	34.3	37.6	61.7	65.7	51.6	91.1	0.1*s*	61.6
Ours (geo-only)	D	x	77.5	86.8	36.5	75.3	27.0	37.5	64.0	66.8	48.7	87.8	**14.5***ms*	60.8
Ours (fused)	D+I	x	71.8	85.3	44.8	76.2	27.0	33.4	68.3	70.0	51.8	88.6	**14.5***ms*	61.7

Table 1: Quantitative detection results on validation set of SUN RGB-D with v1 annotation. The metric is mean average precision (mAP) with 25% 3D-IoU threshold over the 10 most common object classes. The upper half of the table shows results of methods with extra data and the lower half the methods without extra data. Our models marked with * are pretrained on ScanNet frame-level dataset. Some values are absent because they are not reported in the original publications. **Bold**: the best result. ED: extra data. P: raw point clouds. I: color images. V: voxels. D: depth maps.

Methods	mAP(%)
VoteNet	44.0
Ours (geo)	50.8
Ours (fused)	51.2

Table 2: Detection results on ScanNet frame-level detection task.

Methods	Time per Frame (*ms*)				Size (MB)
	Backbone	Head	NMS	Total	
VoteNet [29]	-	-	-	100	11.7
VoteNet	59.1	6.6	0.3	66.0	11.7
Ours (geo)	7.7	6.6	0.3	14.5	67.4
Ours (fused)	7.7	6.6	0.3	14.5	67.4

Table 3: Speed and model size of our models and the baseline. Pre-processing is excluded since it runs in parallel on the CPU.

combining the absolute depth and using RGB fusion brings further improvement (0.4% and 0.9%), respectively.

3D GIoU-Loss. As discussed in [53], object detection benefits from IoU-Loss since it unifies the optimization target and the evaluation metric. To clarify the influence of 3D-GIoU loss, we remove it and train 2.5D-VoteNet with the same loss function as [29]. As shown in Tab. 4, by using 3D-GIoU loss our geometry-only network gains 1.8% improvement, whereas the fused version gains 1.1%. However, even without the advanced loss function, our networks outperform the baseline VoteNet and reach comparable results with SOTA methods.

Fusion with RGB Images. We test different fusion strategies, including directly concatenating normalized depth maps to RGB images as the forth channel, early fusion with an extra Conv layer to accept RGB values (see Fig. 2) and late fusion with two backbones. By the late fusion, the RGB backbone is based on a pre-trained ResNet-34 and has similar up-sampling layers as in Fig. 2. With such two-stream structure, the network learns geometrical and color features with respective backbones. Learned features are then concatenated

First Layers	Input	w/o GIoU	w/ GIoU
Conv	D	55.0	56.1
RDConv	D	58.7	60.4
Conv + RDConv	D	59.0	60.8
Conv + RDConv	D+I	**60.6**	**61.7**

Table 4: Impact of first layers and GIoU-Loss on the detection results on SUN RGB-D. First layers mean layers which directly use RGB images or depth maps as input. D: depth maps. I: RGB images.

Fusion	First Layers	mAP (%)
Geo-only	RDConv + Conv	60.8
RGB-only	Conv	53.3
Naive Concat.	Conv	59.9
Early Fusion	RDConv + Conv	**61.7**
Late Fusion	RDConv + Conv	54.5

Table 5: Detection results with different fusion strategies on SUN RGB-D. Early Fusion is our proposed strategy.

to build feature carriers. To balance the learning rate in different modalities, we use multi-tower training [42] for late fusion, following [30]. Also, we show results with each modality alone. As shown in Tab. 5, the simple and efficient early fusion strategy brings the best result (61.7% mAP). Also, the geometry-only model brings better results than naive concatenation (60.8% vs. 59.9% mAP), which proves the effectiveness of RDConv. Interestingly, the late fusion doesn't generate good results (54.5% mAP). It's probably due to over-fitting, as the two-stream structure requires more data to train.

Reference Depth. RDConv uses the masked mean average depth in each sliding window as reference. Here, we further test the mean and max value of each sliding window as reference. Moreover, depth maps in real-world may contain more noise than SUN RGB-D dataset [39]. To validate the robustness of our networks, we add extra bad pixels in this experiment. Fig. 3 shows that the three choices deliver similar results when no extra bad pixels are added. However, the mean value brings slightly better result than the max value and center value, when depth maps have more bad pixels. We believe that our models are less sensitive to noise in data, if mean value is used as reference depth in RDConv.

Kernel Size. We use 7×7 kernel for RDConv layer. To clarify the impact of the kernel size, we train models with kernel sizes of 3×3, 5×5, 7×7 and 9×9, respectively. We don't use absolute depth in this experiment, in order to emphasize the contribution of RDConv. As shown in Fig. 4, the 3×3 kernel generate 59.9% mAP on SUN RGB-D dataset. The value increases to 60.0% and 60.4% with 5×5 and 7×7 kernel, respectively. Larger kernel (9×9) doesn't bring more improvement.

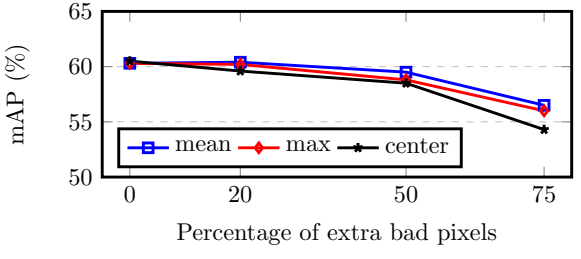

Figure 3: Performance of different reference depths on SUN RGB-D dataset. Absolute depth is not used, in order to emphasize the contribution of RDConv.

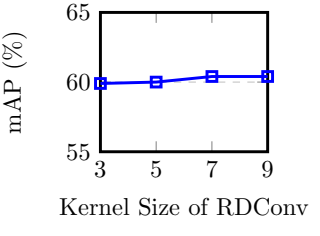

Figure 4: Performance of RDConv with different kernel sizes on SUN RGB-D.

Model Size and Speed. For fair comparison, we adopt the open source code of VoteNet and

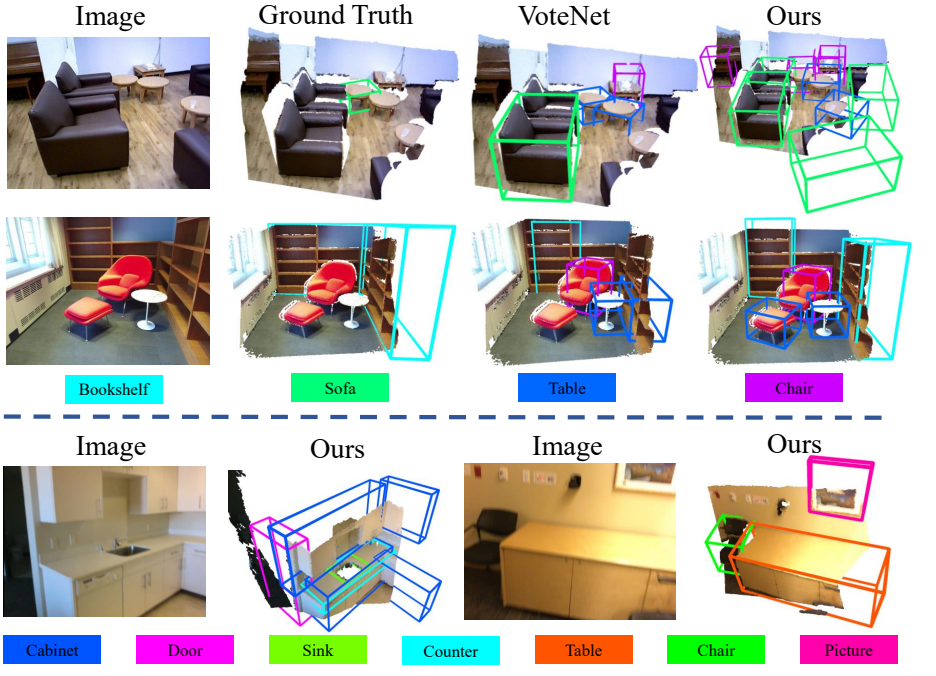

Figure 5: Qualitative results. Upper: comparison of 2.5D-VoteNet and VoteNet [29] on SUN RGB-D dataset [39]. Lower: our detection results on ScanNet [8] frames. Note that SUN RGB-D v1 annotation contains noisy and misaligned labels. All models use solely geometry information. Better viewed with color.

measure the model speed with the same setup in Sec. 4. Due to different hardware and setup, the run-time is shorter than in the publication. As shown in Tab. 3, our models are 4.5 times faster than the baseline, thanks to the efficient 2D backbone. Note the speed of our geo-only and fused models are the same, since the difference is too small to be measured. Also, the fused version has one more Conv layer than the geo-only model and the layer has only ∼5K parameters. To validate our pipeline on mobile devices, we implement 2.5D-VoteNet on an Nvidia Jetson Xavier NX. The Pytorch model runs with 148*ms* per frame (∼7 FPS).

Qualitative Results. As qualitative results, the detected boxes from 2.5D-VoteNet (geo-only version for fair comparison) and VoteNet are illustrated in Fig. 5.

6 Conclusion

In this work we have introduced 2.5D-VoteNet: a simple, powerful and efficient method for real-time 3D object detection. Our models reach state-of-the-art performance and show significant speed improvement over previous point cloud based methods. Our depth map based pipeline is promising, as it allows the application of 2D structures and technique in 3D detection tasks. Due to the lack of annotated data, 3D detectors are usually hard to train and suffer over-fitting. In future work we intend to exploit the self-supervised pretraining for our backbone.

Acknowledgement

This work is financed by Baden-Württemberg Stiftung gGmbH.

References

[1] Syeda Mariam Ahmed and Chee Meng Chew. Density-based clustering for 3d object detection in point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[2] Alex Bewley, Pei Sun, Thomas Mensink, Dragomir Anguelov, and Cristian Sminchisescu. Range conditioned dilated convolutions for scale invariant 3d object detection. *CoRR*, abs/2005.09927, 2020. URL https://arxiv.org/abs/2005.09927.

[3] Jintai Chen, Biwen Lei, Qingyu Song, Haochao Ying, Danny Z. Chen, and Jian Wu. A hierarchical graph network for 3d object detection on point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[4] Shuhan Chen and Yun Fu. Progressively guided alternate refinement network for RGB-D salient object detection. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 520–538, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58598-3.

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

[6] H. Chiang, Y. Lin, Y. Liu, and W. H. Hsu. A unified point-based framework for 3d segmentation. In *2019 International Conference on 3D Vision (3DV)*, pages 155–163, 2019. doi: 10.1109/3DV.2019.00026.

[7] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3075–3084, June 2019.

[8] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. ScanNet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017.

[9] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[10] Hongyuan Du, Linjun Li, Bo Liu, and Nuno Vasconcelos. SPOT: Selective point cloud voting for better proposal in point cloud object detection. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 230–247, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58621-8.

[11] JunYoung Gwak, Christopher Choy, and Silvio Savarese. Generative sparse detection networks for 3d single-shot object detection. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 297–313, Cham, 2020. Springer International Publishing.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, June 2016.

[13] Ji Hou, Angela Dai, and Matthias Niessner. 3D-SIS: 3d semantic instance segmentation of RGB-D scans. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4421–4430, June 2019.

[14] Tengteng Huang, Zhe Liu, Xiwu Chen, and Xiang Bai. EPNet: Enhancing point features with image semantics for 3d object detection. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 35–52, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58555-6.

[15] Wei Ji, Jingjing Li, Miao Zhang, Yongri Piao, and Huchuan Lu. Accurate RGB-D salient object detection via collaborative learning. In *ECCV*, 2020.

[16] Artem Komarichev, Zichun Zhong, and Jing Hua. A-CNN: Annularly convolutional neural networks on point clouds. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7421–7430, June 2019.

[17] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. 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, 2018. doi: 10.1109/IROS.2018.8594049.

[18] Jean Lahoud and Bernard Ghanem. 2d-driven 3d object detection in RGB-D images. In *The IEEE International Conference on Computer Vision (ICCV)*, pages 4622–4630, Oct 2017.

[19] Alex H. Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. PointPillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[20] Bo Li, Tianlei Zhang, and Tian Xia. Vehicle detection from 3d lidar using fully convolutional network. *CoRR*, abs/1608.07916, 2016. URL http://arxiv.org/abs/1608.07916.

[21] Chongyi Li, Runmin Cong, Yongri Piao, Qianqian Xu, and Chen Change Loy. RGB-D salient object detection with cross-modality modulation and selection. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 225–241, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58598-3.

[22] Gongyang Li, Zhi Liu, Linwei Ye, Yang Wang, and Haibin Ling. Cross-modal weighting network for RGB-D salient object detection. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 665–681, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58520-4.

[23] Zhidong Liang, Ming Zhang, Zehan Zhang, Xian Zhao, and Shiliang Pu. RangeRCNN: Towards fast and accurate 3d object detection with range image representation. *CoRR*, abs/2009.00206, 2020. URL https://arxiv.org/abs/2009.00206.

[24] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. URL http://arxiv.org/abs/1405.0312.

[25] Zhijian Liu, Haotian Tang, Yujun Lin, and Song Han. Point-Voxel CNN for efficient 3d deep learning. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32, pages 965–975. Curran Associates, Inc., 2019.

[26] Ao Luo, Xin Li, Fan Yang, Zhicheng Jiao, Hong Cheng, and Siwei Lyu. Cascade graph neural networks for RGB-D salient object detection. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 346–364, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58610-2.

[27] 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/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[28] Charles R. Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J. Guibas. Frustum PointNets for 3d object detection from RGB-D data. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 918–927, June 2018.

[29] Charles R. Qi, Or Litany, Kaiming He, and Leonidas J. Guibas. Deep Hough voting for 3d object detection in point clouds. In *The IEEE International Conference on Computer Vision (ICCV)*, pages 9277–9286, October 2019.

[30] Charles R. Qi, Xinlei Chen, Or Litany, and Leonidas J. Guibas. ImVoteNet: Boosting 3d object detection in point clouds with image votes. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4404–4413, June 2020.

[31] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. PointNet++: Deep hierarchical feature learning on point sets in a metric space. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5099–5108. Curran Associates, Inc., 2017.

[32] Zhile Ren and Erik B. Sudderth. Three-dimensional object detection and layout prediction using clouds of oriented gradients. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1525–1533, June 2016.

[33] 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 (CVPR)*, June 2020.

[34] Weijing Shi and Raj Rajkumar. Point-GNN: Graph neural network for 3d object detection in a point cloud. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[35] Martin Simon, Stefan Milz, Karl Amende, and Horst-Michael Gross. Complex-YOLO: Real-time 3d object detection on point clouds. *CoRR*, abs/1803.06199, 2018. URL http://arxiv.org/abs/1803.06199.

[36] Martin Simon, Karl Amende, Andrea Kraus, Jens Honer, Timo Samann, Hauke Kaulbersch, Stefan Milz, and Horst Michael Gross. Complexer-YOLO: Real-time 3d object detection and tracking on semantic point clouds. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019.

[37] Shuran Song and Jianxiong Xiao. Sliding shapes for 3d object detection in depth images. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 634–651, Cham, 2014. Springer International Publishing. ISBN 978-3-319-10599-4.

[38] Shuran Song and Jianxiong Xiao. Deep sliding shapes for amodal 3d object detection in RGB-D images. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 808–816, June 2016.

[39] Shuran Song, Samuel P. Lichtenberg, and Jianxiong Xiao. SUN RGB-D: A RGB-D scene understanding benchmark suite. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 567–576, June 2015.

[40] H. Thomas, C. R. Qi, J. Deschaud, B. Marcotegui, F. Goulette, and L. Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6410–6419, 2019. doi: 10.1109/ICCV.2019.00651.

[41] Jonas Uhrig, Nick Schneider, Lukas Schneider, Uwe Franke, Thomas Brox, and Andreas Geiger. Sparsity invariant cnns. In *2017 International Conference on 3D Vision (3DV)*, pages 11–20, 2017. doi: 10.1109/3DV.2017.00012.

[42] Weiyao Wang, Du Tran, and Matt Feiszli. What makes training multi-modal classification networks hard? In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12695–12705, June 2020.

[43] Weiyue Wang and Ulrich Neumann. Depth-aware CNN for RGB-D segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[44] Qian Xie, Yu-Kun Lai, Jing Wu, Zhoutao Wang, Yiming Zhang, Kai Xu, and Jun Wang. MLCVNet: Multi-level context votenet for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[45] Saining Xie, Jiatao Gu, Demi Guo, Charles R. Qi, Leonidas Guibas, and Or Litany. PointContrast: Unsupervised pre-training for 3d point cloud understanding. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 574–591, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58580-8.

[46] Y. Xing, J. Wang, X. Chen, and G. Zeng. 2.5D convolution for RGB-D semantic segmentation. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 1410–1414, 2019. doi: 10.1109/ICIP.2019.8803757.

[47] Yajie Xing, Jingbo Wang, and Gang Zeng. Malleable 2.5d convolution: Learning receptive fields along the depth-axis for RGB-D scene parsing. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 555–571, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58529-7.

[48] Danfei Xu, Dragomir Anguelov, and Ashesh Jain. PointFusion: Deep sensor fusion for 3d bounding box estimation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 244–253, June 2018.

[49] 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 (CVPR)*, June 2018.

[50] Zaiwei Zhang, Bo Sun, Haitao Yang, and Qixing Huang. H3DNet: 3d object detection using hybrid geometric primitives. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 311–329, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58610-2.

[51] Jia-Xing Zhao, Yang Cao, Deng-Ping Fan, Ming-Ming Cheng, Xuan-Yi Li, and Le Zhang. Contrast prior and fluid pyramid integration for RGBD salient object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[52] Xiaoqi Zhao, Lihe Zhang, Youwei Pang, Huchuan Lu, and Lei Zhang. A single stream network for robust and real-time RGB-D salient object detection. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 646–662, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58542-6.

[53] D. Zhou, J. Fang, X. Song, C. Guan, J. Yin, Y. Dai, and R. Yang. IoU loss for 2d/3d object detection. In *2019 International Conference on 3D Vision (3DV)*, pages 85–94, 2019. doi: 10.1109/3DV.2019.00019.

[54] Yin Zhou and Oncel Tuzel. VoxelNet: End-to-end learning for point cloud based 3d object detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4490–4499, June 2018.