H3DNet: 3D Object Detection Using Hybrid Geometric Primitives Supplemental Material

Zaiwei Zhang[1], Bo Sun*[1], Haitao Yang*[1], and Qixing Huang[1]

The University of Texas at Austin, Austin, Texas, USA, 78710

1 Introduction

This supplemental material provides the proof of Proposition 1 in Section 2, additional details on network architecture and loss functions in Section 3, more analysis and experiment results on geometric primitive prediction in Section 4, and more analysis and experiment results on 3D object detection in Section 5.

2 Proof of Proposition 1

We show the proof of Proposition 1 here.

With chain rule and equation (3) in the main paper, we can directly get:

$$\frac{\partial l_m}{\partial \Theta} = 2(\boldsymbol{x}^{\star}_{\Theta} - \boldsymbol{x}^{gt})^T \cdot \frac{\partial \boldsymbol{x}^{\star}_{\Theta}}{\partial \Theta}. \tag{1}$$

Since $\boldsymbol{x}^{\star}$ is the local minimum of $f_{\Theta}(\boldsymbol{x})$, we have:

$$\frac{\partial f_{\Theta}(\boldsymbol{x}^{\star})}{\partial \boldsymbol{x}} = 0. \tag{2}$$

Compute the derivatives of both sides w.r.t. Θ, i.e.

$$\frac{\partial^2 f_{\Theta}(\boldsymbol{x}^{\star})}{\partial^2 \boldsymbol{x}} \cdot \frac{\partial \boldsymbol{x}^{\star}}{\partial \Theta} + \frac{\partial^2 f_{\Theta}(\boldsymbol{x}^{\star})}{\partial \boldsymbol{x} \partial \Theta} = 0, \tag{3}$$

which leads to the equation (4) in the main paper.

3 Details on network architecture and loss functions

3.1 Network architecture details

In the main paper, we mentioned that there are three modules in H3DNet: geometric primitive module, proposal generation module, and classification and refinement module. We will discuss each module in detail.

The geometric primitive module first uses a tower of multiple backbone networks to extract down-sampled per-point feature, as shown in Figure 1. The

* Equal Contribution

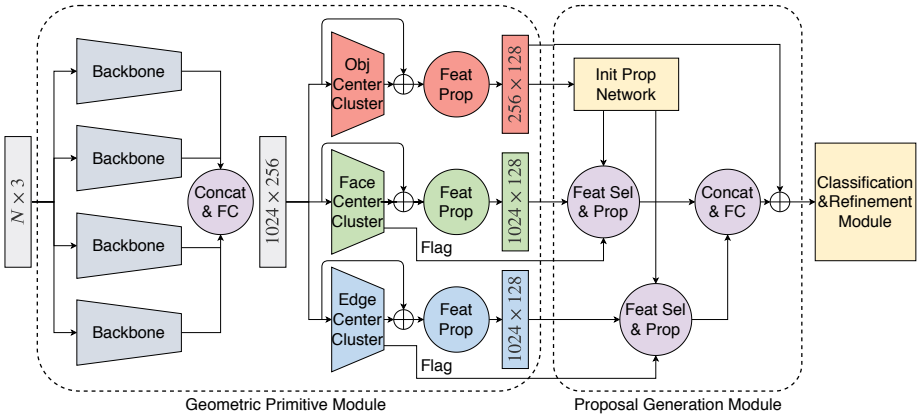

Fig. 1: The pipeline of H3DNet. N represents the number of points of input point clouds, and we use 40000 for both datasets.

backbone network, which is based on PointNet++ [7], was borrowed from [5], and the same network configurations are used in our implementation. For all four backbone networks, we use the same index to sample 1024 points from input point clouds with 40000 points. We then concatenate the features for each point and then use two fully connected layers to reduce the feature dimension to 256. This hybrid feature then feeds into a cluster network, which contains three fully connected layers to predict an offset vector between each point and its corresponding center, i.e., object center, face center, and edge center. For face and edge primitive, we also predict a flag that indicates whether a point is close to a primitive or not. The cluster network also produces a residual feature vector, which will be added to the input feature vector. Finally, we use a set abstraction layer [7], followed by four layers of multilayer perceptron (MLP) after the max-pooling in each local region to propagate features. For object centers, we sub-sampled 256 points for initial proposal generation, using furthest-point-sampling. For face and edge centers, we use the propagated features to predict a point-wise offset vector to refine the center prediction, and a point-wise semantic label to add semantic information in features of geometric primitives.

As shown in Figure 1, we then use three layers of MLP to generate initial object proposals. We use the same configuration as in [5]. As mentioned in the main paper, we then associate each initial object proposal with an overcomplete set of geometric primitives based on the local minimums of the distance function. However, the detected geometric primitives are firstly selected with the predicted flag, which indicates whether a point is close to a primitive or not. Again, we use a set abstraction layer [7], followed by four layers of multilayer perceptron (MLP) after the max-pooling in each local region (i.e., a query ball with radius 0.5m), to propagate features between the predicted geometric primitives and the corresponding primitives of an object proposal. The propagated features are then concatenated and fed into a two-layer MLP for the final object proposals.

The last module is the classification and refinement module. It contains three layers of MLP. We add the feature generated from the proposal module with the object center feature generated in the primitive module, and feed it to the last module to acquire the final object proposal, including an object indicator, offset vectors to refine the BB center, BB size, and BB orientation, and a semantic label. The object indicator is used to determine whether an object exists in the scene or not.

3.2 Loss function details

As mentioned in the main paper, the network is trained end-to-end with a multi-task loss function with five major objective terms. We will discuss each objective term in detail.

$$l_g = l_{vote} + \lambda_1 l_{flag} + \lambda_2 l_{res} + \lambda_3 l_{sem} \tag{4}$$

l_g trains the geometric primitive module. Each primitive has its own objective. For object center offset, face center offset and edge center offset prediction, we adopt the same voting loss defined in [5]. As shown in equation 4, for face and edge center, we add l_{flag} for flag prediction, l_{res} for point-wise center offset prediction (i.e. used in center refinement), and l_{sem} for point-wise center semantic label prediction. We use a L1 loss, defined in [5], for l_{res}, and a standard cross-entropy loss for l_{flag} and l_{sem}. For equation 4, we weight the losses so that they are in similar scales with $\lambda_1 = 3$, $\lambda_2 = 0.1$, and $\lambda_3 = 0.1$.

l_p trains the proposal module θ_p, which contains an objectness loss, a 3D bounding box estimation loss, and a semantic classification loss for initial object proposal generation. We adopt the same loss function defined in [5]. l_f is the distance function defined in the main paper. l_c is a standard cross-entropy loss, which trains the classification sub-network, and l_o contains a cross-entropy loss for semantic label prediction, and an L2 regression loss for BB center, BB size, and BB orientation offset vector prediction. In our experiments, we set the trade-off parameters mentioned in the main paper, λ_g, λ_p, λ_f, λ_c, and λ_o to 1.

3.3 Training details

Our network is implemented in PyTorch and optimized using Adam. The batch size is 8 and the number of epochs is 360. For ScanNet, the learning rate is initialized with 1e-2 and decreases by 10 times after 80, 140, 200, 240 epochs respectively. The learning rate of SUN RGB-D starts with 1e-3 and decreases by 10 times after 160, 220, 260 epochs respectively.

4 Geometric primitive prediction results and analysis

4.1 Dataset Statistics

For an object with a 3D bounding box label, its maximum number of boundary faces is 6, and the maximum number of boundary edges is 12. In a real 3D

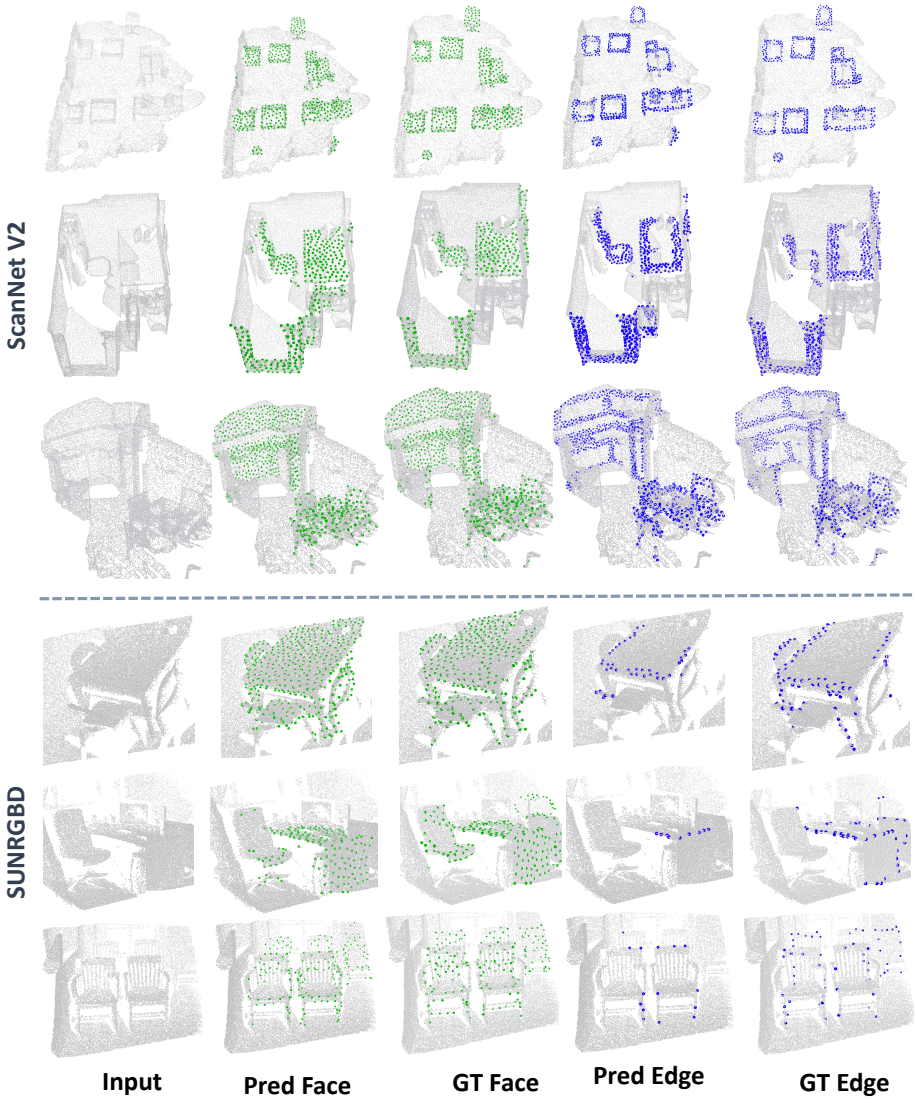

Fig. 2: Qualitative examples for detected geometric primitives (face, edge).

Table 1: Average number of edges and faces labelled per object in the ScanNet training dataset for different categories.

type	cab	bed	chair	sofa	tabl	door	wind	bkshf	pic	cntr	desk	curt	fridg	showr	toil	sink	bath	ofurn	Avg
Face	1.96	3.99	1.74	3.69	2.61	1.71	1.84	3.02	0.75	2.84	2.85	1.97	2.23	2.04	2.27	1.24	4.30	1.63	2.37
Edge	5.60	7.10	6.33	7.48	7.64	3.76	5.19	6.89	2.80	6.90	7.62	5.40	6.17	4.62	7.95	6.23	9.72	5.03	6.25

Table 2: Average number of edges and faces labelled per object in the SUN RGB-D training dataset for different categories.

type	bathtub	bed	bkshf	chair	desk	drser	nigtstd	sofa	table	toilet	Avg
Face	4.42	4.22	2.21	3.61	3.64	2.89	1.04	4.27	3.57	4.21	3.41
Edge	3.04	2.07	1.50	1.80	3.12	2.44	1.16	1.93	2.83	1.91	2.18

scan, some faces or edges are not visible due to occlusion or irregular-shaped objects. As shown in Table 1 and 2, we can see that on both datasets, there are dense labels for faces and edges. However, we can see that the edge labels per object in SUN RGB-D is significantly fewer than in ScanNet. Based on our observation, labels of the 3D bounding boxes in SUN RGB-D are less accurate than in ScanNet, and without per point instance labels, it is much more difficult to generate accurate and dense labels in SUN RGB-D.

4.2 Qualitative Results

We show some qualitative examples for detected geometric primitives in Figure 2. For better visualization purposes, we highlight the detected points if the predicted flag is valid. Most of the examples show that our model performs reasonably well on geometric primitive detection. However, the predictions on edges in SUN RGB-D are sparse due to the lack of labels in training data.

4.3 Quantitative Analysis

In this section, we provide an empirical analysis of the benefits of different geometric primitives. The primary observations are:

– different geometric primitives are suitable for various object categories;

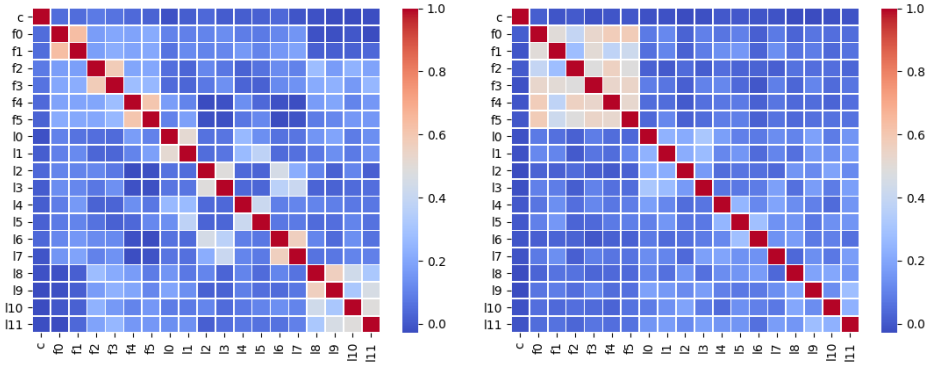

Fig. 3: **Left:** covariance matrix of ScanNet. **Right:** covariance matrix of SUN RGB-D. c represents object center, f0-f6 represent 6 BB face centers, and l0-l11 represent 12 BB edge centers.

Table 3: Prediction accuracy of the location of geometric primitives, i.e. face and edge centers, across different categories for ScanNet. For each target primitive, if there is a prediction within 0.3m, we count it as a correct prediction.

type	cab	bed	chair	sofa	tabl	door	wind	bkshf	pic	cntr	desk	curt	fridg	showr	toil	sink	bath	ofurn
Face	0.89	0.86	0.97	0.91	0.94	0.93	0.83	0.86	0.53	0.93	0.88	0.91	0.99	0.98	0.98	0.98	0.93	0.84
Edge	0.92	0.88	1.00	0.98	1.00	0.91	0.79	0.88	0.81	0.95	0.96	0.81	0.96	0.97	1.00	1.00	0.99	0.76

Table 4: Prediction accuracy of the location of geometric primitives, i.e. face and edge centers, across different categories for SUN RGB-D. For each target primitive, if there is a prediction within 0.3m, we count it as a correct prediction.

type	bathtub	bed	bkshf	chair	desk	drser	nigtstd	sofa	table	toilet
Face	0.72	0.57	0.11	0.83	0.57	0.29	0.15	0.53	0.68	0.91
Edge	0.21	0.04	0.02	0.18	0.29	0.10	0.00	0.04	0.45	0.12

– the bias of the predictions are generally smaller than the variance of the predictions;
– errors in different predictions are mostly uncorrelated.

When aggregating different predictions together, the truncated L2 loss function can prune outlier predictions. Therefore, we obtain a variance reduction and improved prediction accuracy.

Prediction errors under different geometric primtiives. Prediction accuracy of geometric primitives, i.e. face and edge centers, is shown in Table 3 and 4. Since we are predicting an overcomplete set of geometric primitives, we only show the prediction accuracy of detected geometric primitives near the target ground-truth primitives. As shown in Table 3, for most categories, the prediction accuracy of edge center primitives is higher. However, for some categories, like window and curtain, we observe higher accuracy with face center primitive. It shows the different error distributions of BB face centers and BB edge centers, which demonstrate the importance of utilizing multiple geometric primitives.

Table 5: 3D object detection results on SUN RGB-D val dataset. We show per-category results of average precision (AP) with 3D IoU threshold 0.25 as proposed by [9], and mean of AP across all semantic classes. Note that both COG [8] and 2D-driven [4] use room layout context to boost performance. For fair comparison with previous methods, the evaluation is on the SUN RGB-D V1 data.

	RGB	bathtub	bed	bkshf	chair	desk	drser	nigtstd	sofa	table	toilet	mAP.25
DSS[10]	✓	44.2	78.8	11.9	61.2	20.5	6.4	15.4	53.5	50.3	78.9	42.1
COG[8]	✓	58.3	63.7	31.8	62.2	45.2	15.5	27.4	51.0	51.3	70.1	47.6
2D-driven[4]	✓	43.5	64.5	31.4	48.3	27.9	25.9	41.9	50.4	37.0	80.4	45.1
F-PointNet[6]	✓	43.3	81.1	33.3	64.2	24.7	32.0	58.1	61.1	51.1	90.9	54.0
VoteNet [5]	✗	**74.7**	83.0	28.8	75.3	22.0	29.8	62.2	64.0	47.3	90.1	57.7
Ours	✗	73.8	85.6	31.0	**76.7**	**29.6**	**33.4**	**65.5**	**66.5**	50.8	88.2	**60.1**
w\o refine	✗	74.1	**86.4**	**31.3**	76.1	27.1	26.3	57.9	64.9	**51.6**	**89.3**	58.5

Table 6: 3D object detection results on SUN RGB-D val dataset. We show per-category results of average precision (AP) with 3D IoU threshold 0.5 as proposed by [9], and mean of AP. The evaluation is on the SUN RGB-D V1 data.

	RGB	bathtub	bed	bkshf	chair	desk	drser	nigtstd	sofa	table	toilet	mAP.25
VoteNet [5]	✗	**49.9**	47.3	4.6	54.1	5.2	13.6	35.0	41.4	19.7	58.6	32.9
Ours	✗	47.6	**52.9**	**8.6**	**60.1**	**8.4**	**20.6**	**45.6**	**50.4**	**27.1**	**69.1**	**39.0**
w\o refine	✗	48.9	50.6	5.0	55.6	6.3	14.6	32.7	45.1	23.3	60.1	34.2

Table 7: 3D object detection results on ScanNet V2 val dataset. We show per-category results of average precision (AP) with 3D IoU threshold 0.5 as proposed by [9], and mean of AP across all semantic classes with 3D IoU threshold 0.5.

	RGB	cab	bed	chair	sofa	tabl	door	wind	bkshf	pic	cntr	desk	curt	fridg	showr	toil	sink	bath	ofurn	mAP
3DSIS-5[3]	✓	5.73	50.28	52.59	55.43	21.96	10.88	0.00	13.18	0.00	0.00	23.62	2.61	24.54	0.82	71.79	8.94	56.40	6.87	22.53
3DSIS[3]	✗	5.06	42.19	50.11	31.75	15.12	1.38	0.00	1.44	0.00	0.00	13.66	0.00	2.63	3.00	56.75	8.68	28.52	2.55	14.60
Votenet[5]	✗	8.1	76.1	67.2	68.8	42.4	15.3	6.4	28.0	1.3	9.5	37.5	11.6	27.8	10.0	86.5	16.8	78.9	11.7	33.5
Ours	✗	**20.5**	**79.7**	**80.1**	**79.6**	**56.2**	**29.0**	**21.3**	**45.5**	**4.2**	**33.5**	**50.6**	**37.3**	**41.4**	**37.0**	**89.1**	**35.1**	**90.2**	**35.4**	**48.1**
w\o refine	✗	12.4	80.8	69.3	71.8	42.6	19.5	12.3	26.1	2.4	15.7	27.3	32.6	29.5	33.6	79.1	23.0	74.0	18.9	37.3

The prediction accuracy of geometric primitives in SUN RGB-D is significantly lower than in ScanNet, especially for edge center labels. This is again caused by sparse and inaccurate labels in training data.

Figure 4 shows the prediction errors of different geometric primitives under four categories of the test set of ScanNet v2 dataset. We can see that the error patterns are different when varying the categories. This again shows the benefits of having different types of geometric primitives as intermediate supervision. Note that each prediction error is a 3D vector, and we report its norm as the error.

Bias is smaller than the variance. Figure 5 shows that bias and variance of each geometric primitive with respect to the test sets of ScanNet v2 and SUNRBG-D. Here we report the norm of the expecation and the spectrum norm of each 3x3 co-variance matrix. We can see that generally the bias is smaller than the variance. Moreover, this ratio is even smaller on face centers and edge centers than the box center. This shows the usefulness of hybrid geometric primitives.

The reason why bias is smaller than the variance can be understood from the perspective that during training, the training error is generally smaller than the testing error.

Different predictions are mostly uncorrelated. In Figure 3, we visualize the covariance matrix of the error distribution for 19 geometric primitives. For each target geometric primitive, we measure the Euclidean distance to the nearest predicted point and concatenate the results across every object in every testing scene. As shown in Figure 3, the error distributions across all 19 geometric primitives are uncorrelated in ScanNet. Although the error distributions of 6 BB face centers in SUN RGB-D are slightly correlated, other geometric primitives are still uncorrelated.

One interpretation is that in the over-parameterized regime, the optimized network weights are close to the initial network weights [2, 1]. Therefore, if the ini-

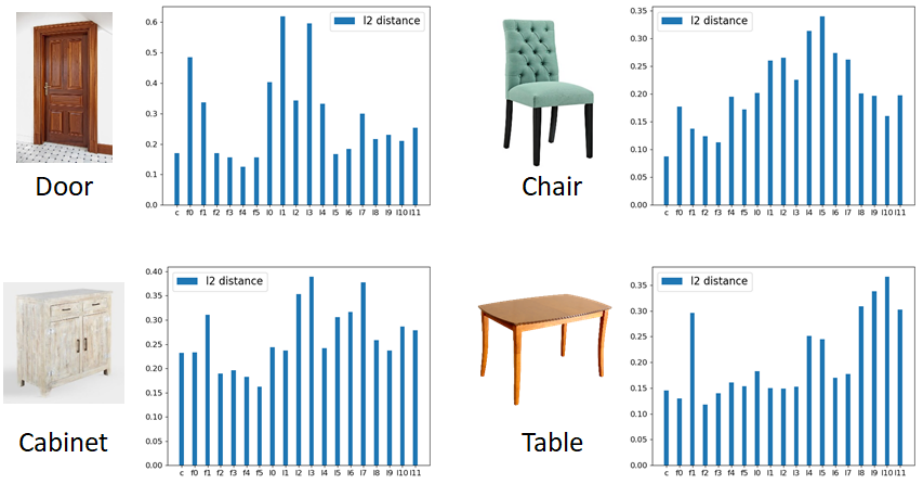

Fig. 4: Prediction errors of different geometric primitives under four different categories of the ScanNet v2 dataset.

tial weights are independent, then the optimized weights are also approximately independent. It follows that different predictions are not strongly correlated. We leave a detailed theoretical analysis for future work.

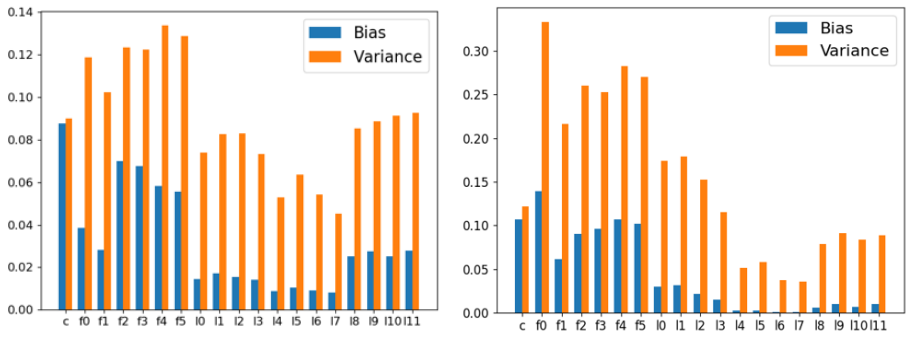

Fig. 5: (Left): magnitudes of bias and variance (square-root) of geometric primitive predictions on ScanNet. (Right): magnitudes of bias and variance (square-root) of geometric primitive predictions on SUNRBGD

Variance reduction and improved accuracy. For simplicity, we focus on analyzing the error of the predicted box center. The analysis of both box parameters are similar. For box center, the prediction is simply a weighted average of

all predictions:

$$\boldsymbol{x}_{pred} = \frac{\boldsymbol{y}_{\text{box}} + \beta_{\text{face}} \sum_{i=1}^{6} \boldsymbol{y}_{\text{face},i} + \beta_{\text{edge}} \sum_{i=1}^{12} \boldsymbol{y}_{\text{edge},i}}{1 + 6\beta_{\text{face}} + 12\beta_{\text{edge}}} \tag{5}$$

where $\boldsymbol{y}_{\text{box}}$ denotes the box center prediction; $\boldsymbol{y}_{\text{face},i}$ denotes the prediction of the i-th face center; $\boldsymbol{y}_{\text{edge},i}$ denotes the prediction of the i-th edge center. Denote the norm of the bias vector of $\boldsymbol{x}_{pred}$ as b_{pred}. It is clear that

$$b_{pred} \leq \frac{b_{\text{box}} + \beta_{\text{face}} \sum_{i=1}^{6} b_{\text{face},i} + \beta_{\text{edge}} \sum_{i=1}^{12} b_{\text{edge},i}}{1 + 6\beta_{\text{face}} + 12\beta_{\text{edge}}}$$

where $b_{\text{face},i}$ and $b_{\text{edge},i}$ are the norms of the bias vectors of $\boldsymbol{y}_{\text{face},i}$ and $\boldsymbol{y}_{\text{edge},i}$, respectively. It is clear that b_{pred} is smaller than the largest bias of each individual prediction.

The variance of $\boldsymbol{x}_{pred}$ is given by

$$V[\boldsymbol{x}_{pred}] = \frac{V[\boldsymbol{y}_{\text{box}}] + \beta_{\text{face}} \sum_{i=1}^{6} V[\boldsymbol{y}_{\text{face},i}] + \beta_{\text{edge}} \sum_{i=1}^{12} V[\boldsymbol{y}_{\text{edge},i}]}{(1 + 6\beta_{\text{face}} + 12\beta_{\text{edge}})^2}$$

Therefore, with suitable chosen trade-off parameters, we obtain a reduction in variance. Combing the fact that the bias is smaller than the variance, x_{pred} is expected to lead to improved accuracy.

5 More Analysis Experiments

5.1 More quantitative results

We show the per-category results on ScanNet with 3D IoU threshold 0.5 in Table 7, and the per-category results on SUN RGB-D with both 3D IoU threshold 0.25 and 0.5 in Table 5 and 6. For accurate object detection, our approach outperforms the baseline approaches significantly. For thin objects in ScanNet, our approach can gain 14.9%, 24.0%, 25.7%, and 27.0% increase on Window, Counter, Curtain, and Shower-curtain. Again, these improvements are achieved by using an overcomplete set of hybrid geometric primitives and their associated features for generating and refining object proposals. Such performance gain can also be observed for SUN RGB-D in Table 6, where our approach performs significantly better on more accurate object detection.

5.2 More qualitative results

We show more qualitative examples of 3D object detection for both datasets in Figure 8 and 9. In Figure 6, we show qualitative comparisons between our approach and the top-performing baseline approach on thin objects. Our method is more accurate and detects more positive thin objects than baseline approaches. We also show some failure cases with our approach in Figure 7, and we summarize the failure patterns in the caption.

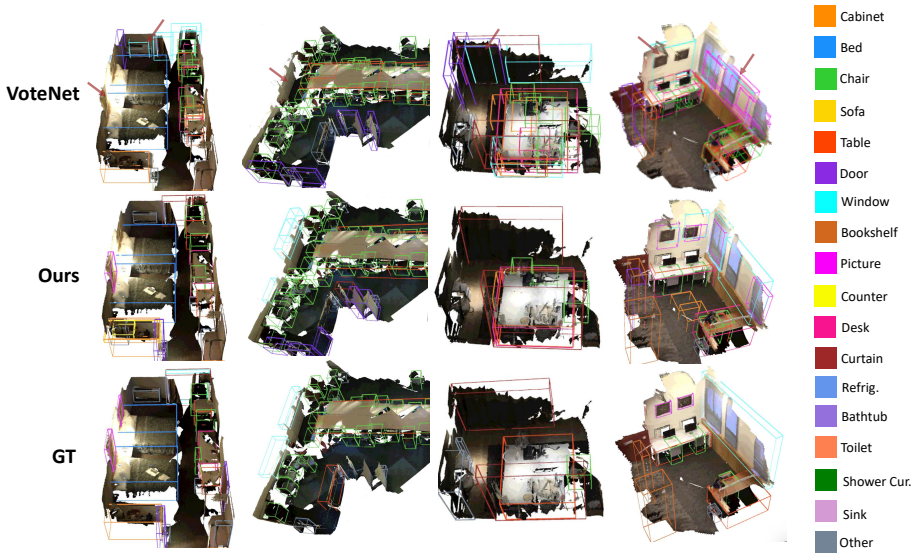

Fig. 6: Qualitative evaluation on thin object detection. Red arrows are used to highlight the thin objects.

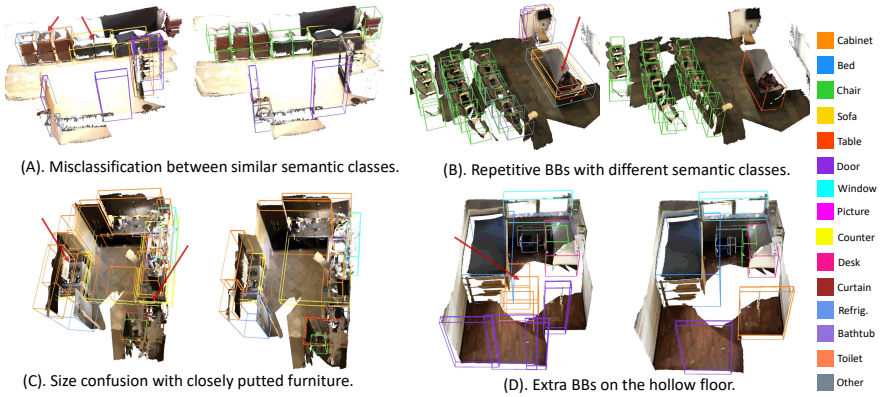

Fig. 7: Samples of failure cases.

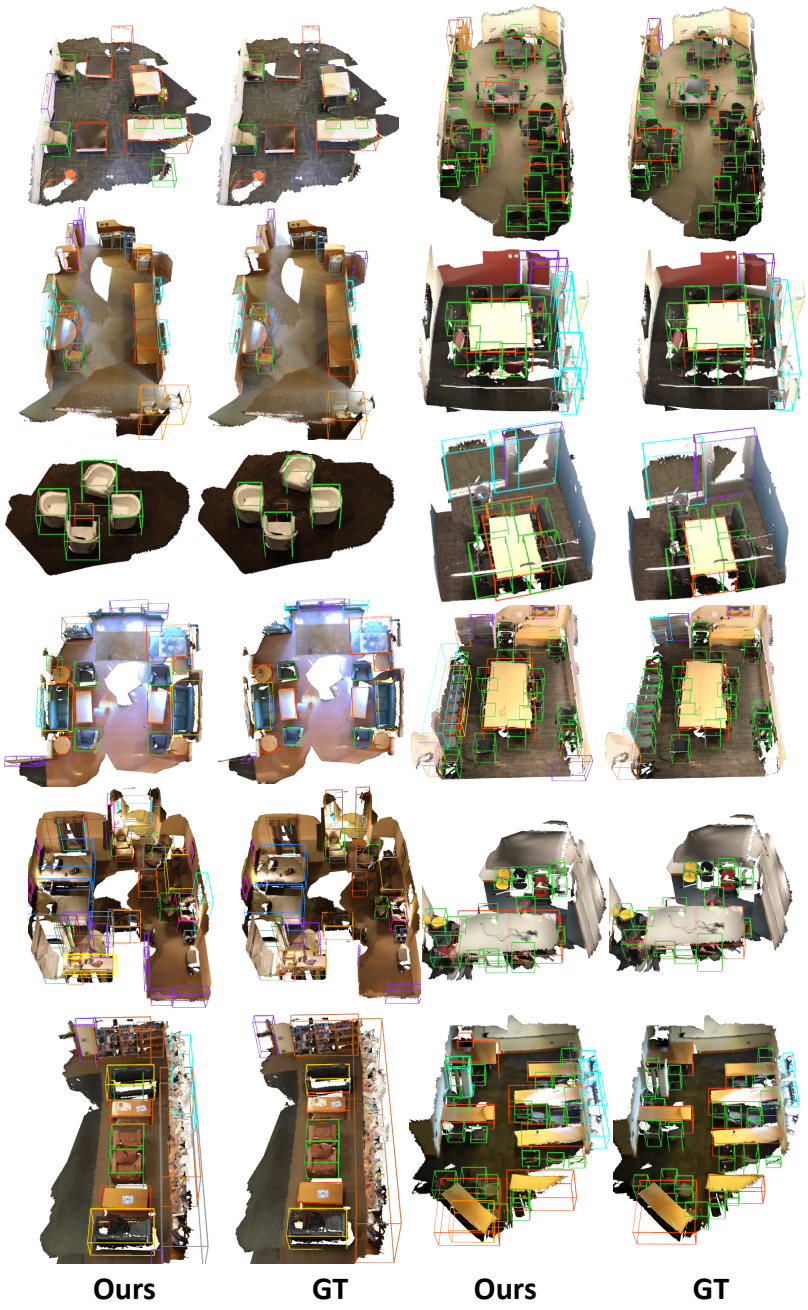

Fig. 8: More qualitative results on ScanNet V2.

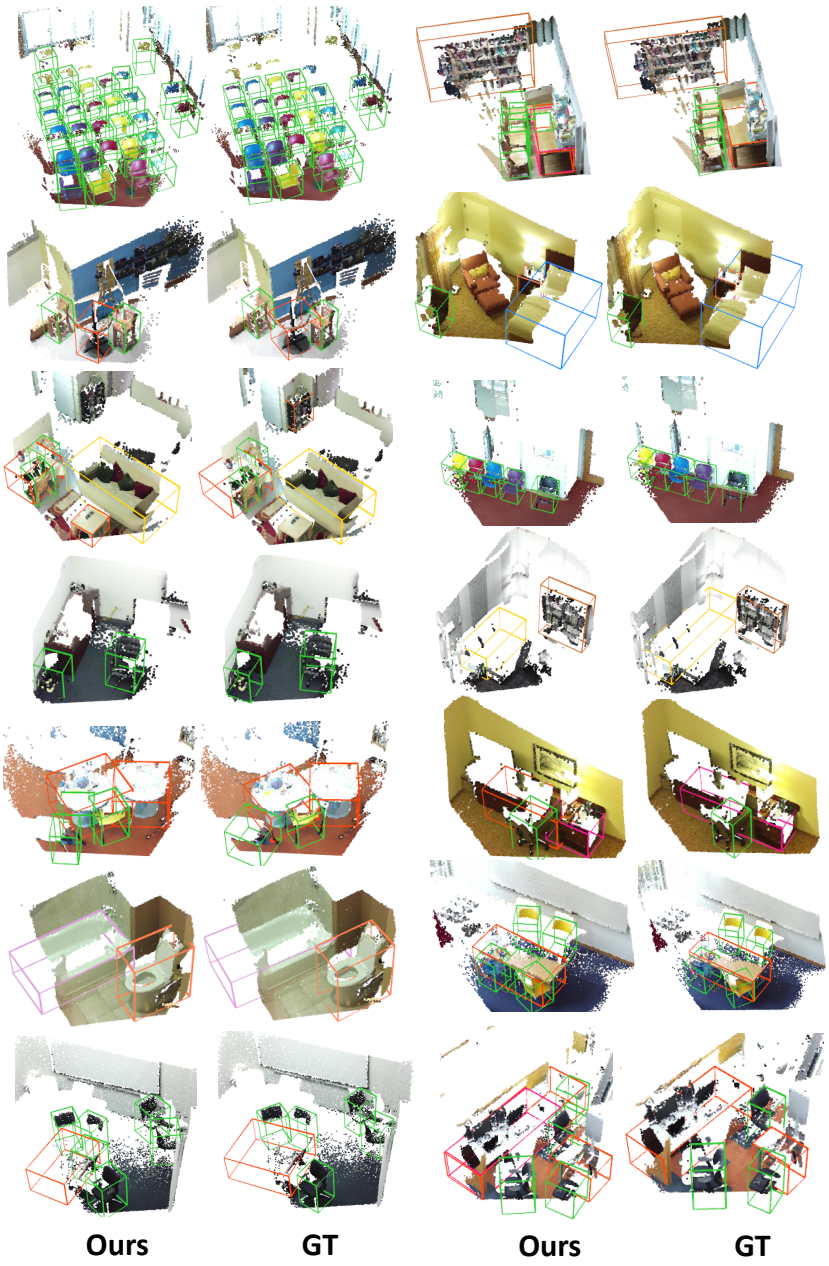

Fig. 9: More qualitative results on SUN RGB-D V1.

References

1. Arora, S., Du, S.S., Hu, W., Li, Z., Wang, R.: Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In: Chaudhuri, K., Salakhutdinov, R. (eds.) Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA. Proceedings of Machine Learning Research, vol. 97, pp. 322–332. PMLR (2019), http://proceedings.mlr.press/v97/arora19a.html
2. Du, S.S., Zhai, X., Póczos, B., Singh, A.: Gradient descent provably optimizes over-parameterized neural networks. In: 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019. OpenReview.net (2019), https://openreview.net/forum?id=S1eK3i09YQ
3. Hou, J., Dai, A., Niessner, M.: 3d-sis: 3d semantic instance segmentation of rgb-d scans. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019)
4. Lahoud, J., Ghanem, B.: 2d-driven 3d object detection in rgb-d images. In: The IEEE International Conference on Computer Vision (ICCV) (Oct 2017)
5. Qi, C.R., Litany, O., He, K., Guibas, L.J.: Deep hough voting for 3d object detection in point clouds. arXiv preprint arXiv:1904.09664 (2019)
6. Qi, C.R., Liu, W., Wu, C., Su, H., Guibas, L.J.: Frustum pointnets for 3d object detection from rgb-d data. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 918–927 (2018)
7. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In: Advances in neural information processing systems. pp. 5099–5108 (2017)
8. Ren, Z., Sudderth, E.B.: Three-dimensional object detection and layout prediction using clouds of oriented gradients. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2016)
9. 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)
10. Song, S., Xiao, J.: Deep sliding shapes for amodal 3d object detection in rgb-d images. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2016)