Appendix

In this section, we provide: (1) algorithm details and ablation studies of Geometric Selective Search (Appendix A); (2) introduction of the shape detection algorithm (Appendix B); (3) additional implementation details (Appendix C); (4) details of the integration of an external object prior (Appendix D); and (5) per-class segmentation results; (6) additional qualitative results (Appendix F).

A. Geometric Selective Search (GSS)

As introduced in the main paper § 3.3, the goal of GSS is to capture all possible object locations in 3D space. We formulate a bottom-up algorithm where the key idea is to utilize the geometric and semantic cues for guiding 3D proposal generation.

A.1. Approach

Given an input point cloud with unoriented normals, we first detect primitive shapes using a region growing based method [22] as detailed in Appendix B. It outputs a set of detected planes with assigned points, *i.e.*, each point is assigned to at most one plane or none.

We then apply hierarchical agglomerative clustering (HAC) to generate the candidate bounding boxes from the detected planes. We first initialize a region set with the detected planes, and then compute the similarity score s between all neighboring regions in the set. Two regions are neighboring if the corresponding convex hull of them overlap. To overcome the artifacts of the point cloud, we randomly jitter the points of each region before computing their convex hull. This technique greatly improves the results in practice as verified in Fig. 4. Once the neighboring relationships and similarity scores are computed, the two most similar regions are grouped into a new region. We then generate an axis-aligned 3D box for the new region as a proposal. New similarity scores are calculated between the resulting region and its neighbors. HAC is repeated until no neighbors can be found or only a single region remains. We provide the detailed pseudo-code in Alg. 3.

In order to pick which two regions $\mathbf{n}_i, \mathbf{n}_j$ to group, we use the similarity score $s(\mathbf{n}_i, \mathbf{n}_j) =$

$$\begin{aligned} &w_1 s_{\text{color}}(\mathbf{n}_i, \mathbf{n}_j) + w_2 s_{\text{size}}(\mathbf{n}_i, \mathbf{n}_j) + \\ &w_3 s_{\text{volume}}(\mathbf{n}_i, \mathbf{n}_j) + w_4 s_{\text{fill}}(\mathbf{n}_i, \mathbf{n}_j) + w_5 s_{\text{seg}}(\mathbf{n}_i, \mathbf{n}_j), \end{aligned} \tag{13}$$

where $w_i \in \{0, 1\} \ \ \forall i \in \{1, \cdots, 5\}$ are binary indicators. Binary weights are used over continuous values to encourage more diverse outputs following [48]. $s_{\text{color}} \in [0, 1]$ measures the color similarity; s_{size} and $s_{\text{volume}} \in [0, 1]$ measure size and volume compatibility and encourage small regions to merge early; $s_{\text{fill}} \in [0, 1]$ measures how well two regions are aligned; and $s_{\text{seg}}(\mathbf{n}_i, \mathbf{n}_j) \in [0, 1]$ measures high-level semantic similarities. We detail each metric next.

Algorithm 3 Geometric Selective Search (GSS)

Input: point cloud $\mathcal{P}$
Output: 3D proposal set $\mathcal{R}$
Detect shapes from $\mathcal{P} \rightarrow$ initial regions $\mathcal{N} = \{\mathbf{n}_1, \mathbf{n}_2, \cdots\}$
Initialize similarity set $\mathcal{S} = \emptyset$, proposal set $\mathcal{R} = \emptyset$
for each neighboring region pair $(\mathbf{n}_i, \mathbf{n}_j)$ **do**
  $\mathcal{S} = \mathcal{S} \cup s(\mathbf{n}_i, \mathbf{n}_j)$ ▷ compute and store similarities
while $\mathcal{S} \neq \emptyset$ **do** ▷ HAC
  Get the most similar pair $s(\mathbf{n}_i, \mathbf{n}_j) = \max(\mathcal{S})$
  Remove similarities regarding $\mathbf{n}_i$: $\mathcal{S} = \mathcal{S} \setminus s(\mathbf{n}_i, *)$
  Remove similarities regarding $\mathbf{n}_j$: $\mathcal{S} = \mathcal{S} \setminus s(\mathbf{n}_j, *)$
  Update region set $\mathcal{N} = \mathcal{N} \setminus \mathbf{n}_i, \mathcal{N} = \mathcal{N} \setminus \mathbf{n}_j$
  Merge and generate new region $\mathbf{n}_k = \mathbf{n}_i \cup \mathbf{n}_j$
  Compute similarity of $\mathbf{n}_k$ and its neighbors in $\mathcal{N}$: $\mathcal{S} = \mathcal{S} \cup \{s(\mathbf{n}_k, \mathbf{n}') : \text{neighbor}(\mathbf{n}_k, \mathbf{n}') = \text{True} \ \forall \mathbf{n}' \in \mathcal{N}\}$
  Add new region to $\mathcal{N} = \mathcal{N} \cup \mathbf{n}_k$
  Generate 3D proposal $\mathcal{R} = \mathcal{R} \cup \text{AxisAlignedBox}(\mathbf{n}_k)$

Color similarity s_{color}. Color is an informative low-level cue to guide the plane grouping process. For each region, we first compute the L1-normalized color histogram following [48]. The similarity score is computed as the histogram intersection:

$$s_{\text{color}}(\mathbf{n}_i, \mathbf{n}_j) = \sum_k \min(b_i^k, b_j^k), \tag{14}$$

where b_i^k, b_j^k are the k-th bin in the color histograms of $\mathbf{n}_i$ and $\mathbf{n}_j$ respectively. Following [48], we use 25 bins for each HSV color channel and 75 in total for one histogram.

Size similarity s_{size} and volume similarity s_{volume}. These two metrics encourage small regions to merge early. This strategy is desirable as it guarantees a bottom-up grouping of parts of different objects at multiple locations in 3D space. It encourages diverse 3D proposals and prevents a single region from absorbing all other regions gradually. We compute size similarity

$$s_{\text{size}}(\mathbf{n}_i, \mathbf{n}_j) = 1 - \frac{\text{size}(\mathbf{n}_i) + \text{size}(\mathbf{n}_i)}{\text{size}(\mathcal{P})}, \tag{15}$$

where $\text{size}(\mathbf{n}_i), \text{size}(\mathbf{n}_j), \text{size}(\mathcal{P})$ are the size of the axis-aligned bounding boxes of region $\mathbf{n}_i, \mathbf{n}_j$ and the whole point cloud. Similarly, volume similarity is defined as:

$$s_{\text{volume}}(\mathbf{n}_i, \mathbf{n}_j) = 1 - \frac{\text{volume}(\mathbf{n}_i) + \text{volume}(\mathbf{n}_i)}{\text{volume}(\mathcal{P})}, \tag{16}$$

where $\text{volume}(\mathbf{n}_i), \text{volume}(\mathbf{n}_j), \text{volume}(\mathcal{P})$ are the volume of the water-tight convex hull of region $\mathbf{n}_i, \mathbf{n}_j$ and the whole point cloud.

Alignment score s_{fill}. This score measures how well two regions fit into each other and encourage merged regions to be cohesive. Essentially, if one region is contained in the other one, they should be merged first to avoid any holes. Meanwhile, a low score means the two regions don't fit very

Class	cabinet	bed	chair	sofa	table	door	window	shelf	picture	counter	desk	curtain	fridge	sc*	toilet	sink	bathtub	other	mean
Unsupervised GSS																			
ABO	0.402	0.414	0.419	0.462	0.432	0.327	0.349	0.469	0.121	0.286	0.365	0.342	0.469	0.421	0.415	0.355	0.325	0.432	0.378
Recall	86.0	97.5	90.4	99.0	91.1	67.0	86.9	100.0	26.1	75.0	92.1	91.0	98.2	96.4	94.8	91.8	77.4	90.9	86.2
GSS																			
ABO	0.449	0.471	0.441	0.437	0.464	0.379	0.388	0.446	0.136	0.366	0.381	0.399	0.501	0.478	0.409	0.365	0.400	0.453	0.409
Recall	90.6	98.8	91.7	98.9	93.7	75.2	89.7	100.0	27.9	88.5	94.5	97.0	96.5	100.0	94.8	92.9	83.8	92.3	89.3

Table 6: Per-class results of GSS proposals. GSS achieves more than 80% recall rate for all classes except picture (27.9%) and door (75.2%), where the plan detection algorithm often fails to differentiate these two objects from the surrounding wall. Here sc* refers to the 'shower curtain' class.

well, and they may form an unnatural region. We compute the alignment score:

$$s_{\text{fill}}(\mathbf{n}_i, \mathbf{n}_j) = 1 - \frac{\text{size}(\mathbf{n}_i \cup \mathbf{n}_j) - \text{size}(\mathbf{n}_i) - \text{size}(\mathbf{n}_i)}{\text{size}(\mathcal{P})}, \tag{17}$$

where $\mathbf{n}_i \cup \mathbf{n}_j$ means the union of two regions, and the other numbers are identical to the ones used for the computation of s_{color}.

Semantic similarity s_{seg}. The above four metrics are mainly low-level geometric cues. GSS can also utilize high-level semantic information, *i.e.*, weakly-supervised segmentation prediction. For each region, we first infer the segmentation mask from $\mathbf{S}_{\text{seg}}$ using the inference procedure described in § 4. We then take the most likely class assignment for each point in the region and compute an L1-normalized histogram over classes for that region. The similarity score is computed as the histogram intersection:

$$s_{\text{seg}}(\mathbf{n}_i, \mathbf{n}_j) = \sum_{c=1}^{C} \min(b_i^c, b_j^c), \tag{18}$$

where b_i^c, b_j^c are the bin of class c in the class histograms.

Post-processing. To remove the redundant proposals, we use several post-processing steps: (1) the proposals are first filtered by a 3D NMS module with an IoU threshold of 0.75; (2) we then remove the largest bounding boxes after NMS as it covers the whole scene rather than certain objects due to the bottom-up nature of HAC; (3) we keep at most 1000 proposals through random sampling.

Diversification strategies. Since a single strategy usually overfits, we adopt multiple strategies to encourage a diverse set of proposals, which will eventually lead to a better coverage of all objects in 3D space. Specifically, we first create a set of complementary strategies, and ensemble their results afterwards. Highly-overlapping redundant proposals are removed though an NMS with IoU threshold of 0.75 and we still keep at most 1000 proposals through random sampling after ensembeling.

Metric	Avg. # boxes	MABO	AR
Single run			
SZ	382.9	0.351	84.1
C	252.0	0.316	70.7
V	366.8	0.367	84.4
F	330.2	0.398	81.8
SG	350.7	0.362	83.9
SZ+V	373.4	0.366	84.5
SZ+SG	369.2	0.353	85.1
V+F	373.3	0.384	85.7
V+SG	385.5	0.362	83.8
SZ+V+SG	377.5	0.391	86.4
V+F+SG	381.6	0.380	84.9
SZ+V+F+SG	369.1	0.387	86.1
Ensembeling			
V+F, SZ+V	712.0	0.378	86.2
V+F, SZ+V+SG	742.9	0.409	89.3

Table 7: GSS results using various similarity metrics. SZ, C, V, F, and SG represent s_{size}, s_{color}, s_{volume}, s_{fill}, and s_{seg} respectively.

A.2. Experiments

In this sub-section we evaluate the proposal quality of GSS and validate the corresponding design choices. We evaluate on the ScanNet validation set and report the two popular metrics: average recall (AR) and mean average best overlap (MABO) across all classes. In addition, we also report the average number of boxes of each scene.

We first examine each similarity metric and their combinations in Tab. 7. We first evaluate each single similarity and report their results in the top 5 rows, where we find size, volume, and segmentation metric to work much better than color and fill similarity. Tab. 7 also reports the results of different combined metrics. Combining multiple similarity metrics often yields better results than using each single similarity. The best result is achieved using the combination of size, volume, and segmentation similarities.

In practice, we find that ensembling the results of multiple runs using different similarity metrics further improves

the results as shown in Tab. 7 bottom. We provide the results of an unsupervised version (V+F, SZ+V) and the complete version (V+F, SZ+V+SG). Comparing these two methods, we find that introducing segmentation similarity is beneficial.

Lastly, we show per-class average best overlap (ABO) and recall rate in Tab. 6. We find that GSS achieves high recall rate ($> 80\%$) for all classes except picture (27.9%) and door (75.2%). This is likely due to the fact that these two objects are often embedded in the wall and hard to differentiate.

A.3. Qualitative results

Fig. 3 illustrates several representative examples of the generated proposals on ScanNet. From left to right, we show the input point cloud, the detected shapes, GSS computed proposals, and the ground-truth boxes. We show all the GSS computed proposals in the top 3 rows where we observe that the computed proposals are mainly around each object in the scene. In the bottom four rows, we show the best overlapping proposals with ground-truth bounding boxes. GSS generates proposals with great recall, and generalizes well to various object classes and complex scenes.

B. Shape detection

In this paper, we detect geometric shapes for two reasons: to be used in the local smoothness loss for segmentation (Eq. (7)), and as input to the GSS algorithm (Appendix A). As introduced in the main paper § 3.3, we adopt a region-growing algorithm [22, 26] for detecting primitive shapes (*e.g.*, planes). The basic idea is to iteratively detect shapes by growing regions from seed points. Specifically, we first choose a seed point and find its neighbors in the point cloud. These neighbors are added to the region if they satisfy the region requirements (*e.g.*, on the same plane), and hence the region grows. We then repeat the procedure for all the points in the region until no neighbor points meet the requirements. In the latter case we start a new region. Region-growing out-performs the popular RANSAC-based methods [38] because 1) it is deterministic; 2) it performs better in the presence of large scenes with fine-grained details; 3) it has higher shape detection recall. Even though it runs slower, we use it as a pre-processing step which won't influence the training speed.

In practice, we use the efficient implementation of The Computational Geometry Algorithms Library (CGAL) [26]. We set the search space to be the 12 nearest neighbors, the maximum distance from the furthest point to a plane to be 12, the maximum accepted angle between a point's normal and the normal of a plane to be 20 degree, and the minimum region size to be 50 points. We refer the reader to CGAL documents [26] for more details.

Representative visualization of the detected planes are provided in Fig. 6 second column from left. The algorithm detects big planes (*e.g.*, floor, table top, wall) with great accuracy and doesn't over segment these regions into small pieces. This is particularly useful for WyPR as the local smoothness loss will enforce the segmentation module to predict consistently within these shapes. For complex objects (*e.g.*, curtain, chair, and bookshelf), this algorithm segments the object regions into small shapes. Such primitive shapes will be used during the proposal generation algorithm GSS to infer the 3D bounding boxes of all objects in the scene.

C. Additional implementation details

In this section, we provide additional implementation details.

C.1. Geometric transformations

We apply geometric transformations in two places: 1) as data-augmentation; 2) for computing cross-transformation consistency losses (Eq. (5) and Eq. (11)) for both tasks.

To augment the input, we first randomly sub-sample 40,000 points as input in each training iteration. We then randomly flip the points in both horizontal and vertical directions with probability 0.5, and randomly rotate them around the upright-axis with $[-5, 5]$ degree. Note that after data augmentation, we only get one point cloud $\mathcal{P}$ as input.

To compute the consistency losses, we further transform the input point cloud using random flipping of both horizontal and vertical directions with probability 0.5, larger random rotation of $[0, 30]$ degrees around the upright-axis, random scaling by a factor within $[0.8, 1.2]$, and point dropout ($p = 0.1$). We denote the resulting point cloud as $\tilde{\mathcal{P}}$, which will be used when computing $\mathcal{L}_{\text{seg}}^{\text{CST}}$ and $\mathcal{L}_{\text{det}}^{\text{CST}}$.

C.2. Backbone

We adopt a PointNet++ network as backbone, which has four set abstraction (SA) layers and two feature propagation (FP) layers. For a fair comparison we use the same backbone network as Qi *et al*. [29]. The input to the backbone is a fix-sized point cloud where we randomly sample 40,000 points from the original scans. The outputs of the backbone network are geometric representations of 1024 points with dimension 3+256 (XYZ+feature dimension).

C.3. Segmentation module

The segmentation module contains two feature propagation (FP) layers which upsample the geometric representations of 1024 points to 2048 and then 40,000 points with the same dimension (3+256) as before. We then use a two-layer MLP with dimension $[256, C]$ as the classifier where

metric	cabinet	bed	chair	sofa	table	door	shelf	desk	curtain	fridge	toilet	sink	bathtub
$\mu_{l:w}$	4.64	1.58	1.29	1.94	1.65	5.74	3.17	1.92	5.78	1.68	1.55	1.29	1.93
$\sigma_{l:w}$	5.81	0.45	0.53	0.54	1.02	3.78	2.07	0.91	3.58	1.16	0.39	0.26	0.42
$\mu_{l:h}$	1.49	2.12	1.16	2.36	3.04	0.61	1.22	2.28	1.40	0.65	1.08	2.14	3.18
$\sigma_{l:h}$	1.01	0.95	0.98	0.57	3.72	0.69	1.11	1.65	1.34	0.19	0.56	0.89	1.67

Table 8: Prior statistics for each class.

C represents the number of classes. The segmentation module outputs a dense semantic prediction for each point in the point cloud.

C.4. Detection module

The detection module first applies a RoI pooling by average-pooling the features of all points within each RoI. The computed RoI features are then fed into three fully-connected layers to get the classification $\mathbf{S}_{\text{cls}}$, objectness $\mathbf{S}_{\text{obj}}$, and final classification logits $\mathbf{S}_{\text{det}}$ respectively.

C.5. Losses

For computing the smoothness regularization $\mathcal{L}_{\text{smooth}}$ in Eq. (7), enumerating all the detected planes in each training iteration is time-consuming and not necessary. We thus randomly sample 10 planes in each iteration, as we find 10 to be the sweet spot balancing training speed and performance. For computing the self-training losses $\mathcal{L}_{\text{seg}}^{\text{SELF}}$ and $\mathcal{L}_{\text{det}}^{\text{SELF}}$, we set the threshold p_1 in Alg. 1 to be 0.1, and p_2 in Alg. 2 to be 0.15. The threshold τ in Alg. 2 is set to 0.25.

D. External prior

WyPR can be further improved by integrating external object priors as shown in Tab. 2 and Tab. 3. We mainly introduce two types of priors as they can be easily computed from external synthetic datasets [6, 57]: the shape prior and the location prior.

For the shape prior, we compute the mean aspect ratio between an object's 3D bounding box length to height ($\mu_{l:h}^c$), and length to width ($\mu_{l:w}^c$) for class $c \in \{1, \cdots, C\}$. Since objects can be of arbitrary pose in 3D space, we set length and width to measure the longer and shorter edge in the XY plane. We also compute the corresponding standard deviations $\sigma_{l:h}^c$ and $\sigma_{l:d}^c$. To use it, we reject proposals whose aspect ratios don't fall within the range $[\mu_{l:h}^c - 2\sigma_{l:h}^c, \mu_{l:h}^c + 2\sigma_{l:h}^c]$ and $[\mu_{l:w}^c - 2\sigma_{l:w}^c, \mu_{l:w}^c + 2\sigma_{l:w}^c]$ for any class $c \in \{1, \cdots, C\}$. We also reject pseudo bounding boxes of ground-truth class c ($R^*[c]$ in Alg. 2) whose aspect ratios don't fall in $[\mu_{l:h}^c - 2\sigma_{l:h}^c, \mu_{l:h}^c + 2\sigma_{l:h}^c]$ and $[\mu_{l:w}^c - 2\sigma_{l:w}^c, \mu_{l:w}^c + 2\sigma_{l:w}^c]$. The computed statistics of each class are shown in Tab. 8. There are certain classes that are missing from the external synthetic datasets [6, 57] such as shower curtain, window, counter, and picture. For these classes, we use the prior of other objects with similar shapes as a replacement. For example, we use the prior of curtain for shower curtain, table for counter, door for window and picture.

The location prior is only applied to the floor class. This prior is of vital importance as floor appears in almost every scene. It becomes a hard class for semantic segmentation as the MIL loss rarely sees any negative examples. Besides, a great portion of points in each scene belongs to the floor. We estimate the floor height as the 1% percentile of all points' heights following Qi *et al*. [29]. We force all the points below floor height to be floor. All the points above this height cannot be floor.

E. Per-class segmentation results

In Tab. 9, we report the per-class IoU on ScanNet. These results are consistent with Tab. 2 in main paper. Compared to prior methods PCAM [53] and MPRM [53], WyPR significantly out-performs them, and greatly improves the performance of some hard classes such as door, counter, and fridge.

F. Additional qualitative results

In Fig. 7, we show the qualitative comparison between ground-truth labels and our (WyPR+prior) prediction. In each row we show the results of both tasks for one scene. We find that WyPR segments and detects certain classes (table in row (a, f), chair in rows (a, b, f), sofa in row (b), bookshelf in row (c, f)) with great accuracy. WyPR also learns to recognize some uncommon objects of the dataset such as toilet and sink in row (d). Moreover, we observe that predicted segmentation mask and bounding boxes are highly consistent, which reflects the effectiveness of the joint-training framework.

Common failure cases for WyPR are partially observed objects (row (b): the window on the left side), ambiguous objects (row (a): picture and wall; row (b, f): sofa and left-most chair). When multiple objects of the same classes are spatially close, WyPR often cannot differentiate them and only predicts one big boxes covering everything (row (a): tow chair on the left side).

Methods	eval.	wall	floor	cabinet	bed	chair	sofa	table	door	window	shelf	picture	counter	desk	curtain	fridge	sc*	toilet	sink	bathtub	other	mIoU
PCAM [53]	train	54.9	48.3	14.1	34.7	32.9	45.3	26.1	0.6	3.3	46.5	0.6	6.0	7.4	26.9	0.0	6.1	22.3	8.2	52.0	6.1	22.1
MPRM [53]	train	47.3	41.1	10.4	43.2	25.2	43.1	21.5	9.8	12.3	45.0	9.0	13.9	21.1	40.9	1.8	29.4	14.3	9.2	39.9	10.0	24.4
WyPR	train	**59.3**	31.5	6.4	**58.3**	31.6	**47.5**	18.3	**17.9**	**36.7**	34.1	6.2	**36.1**	**24.3**	**67.2**	**8.7**	**38.0**	17.9	**28.9**	35.9	8.2	**30.7**
MIL-seg	val	36.4	36.1	**13.5**	37.9	25.1	31.4	9.6	18.3	19.8	33.1	**7.9**	20.3	21.7	32.5	6.4	14.0	7.9	14.7	19.4	8.5	20.7
WyPR	val	**58.1**	33.9	5.6	**56.6**	29.1	**45.5**	19.3	15.2	**34.2**	**33.7**	6.8	**33.3**	22.1	65.6	6.6	**36.3**	18.6	24.5	**39.8**	6.6	29.6
WyPR+prior	val	52.0	**77.1**	6.6	54.3	**35.2**	40.9	**29.6**	9.3	28.7	33.3	4.8	26.6	**27.9**	**69.4**	**8.1**	27.9	**24.1**	**25.4**	32.3	**8.7**	**31.1**

Table 9: 3D semantic segmentation on ScanNet. WyPR outperforms standard baselines and existing state-of-the-art [53] by a margin. Here sc* refers to the 'shower curtain' class.

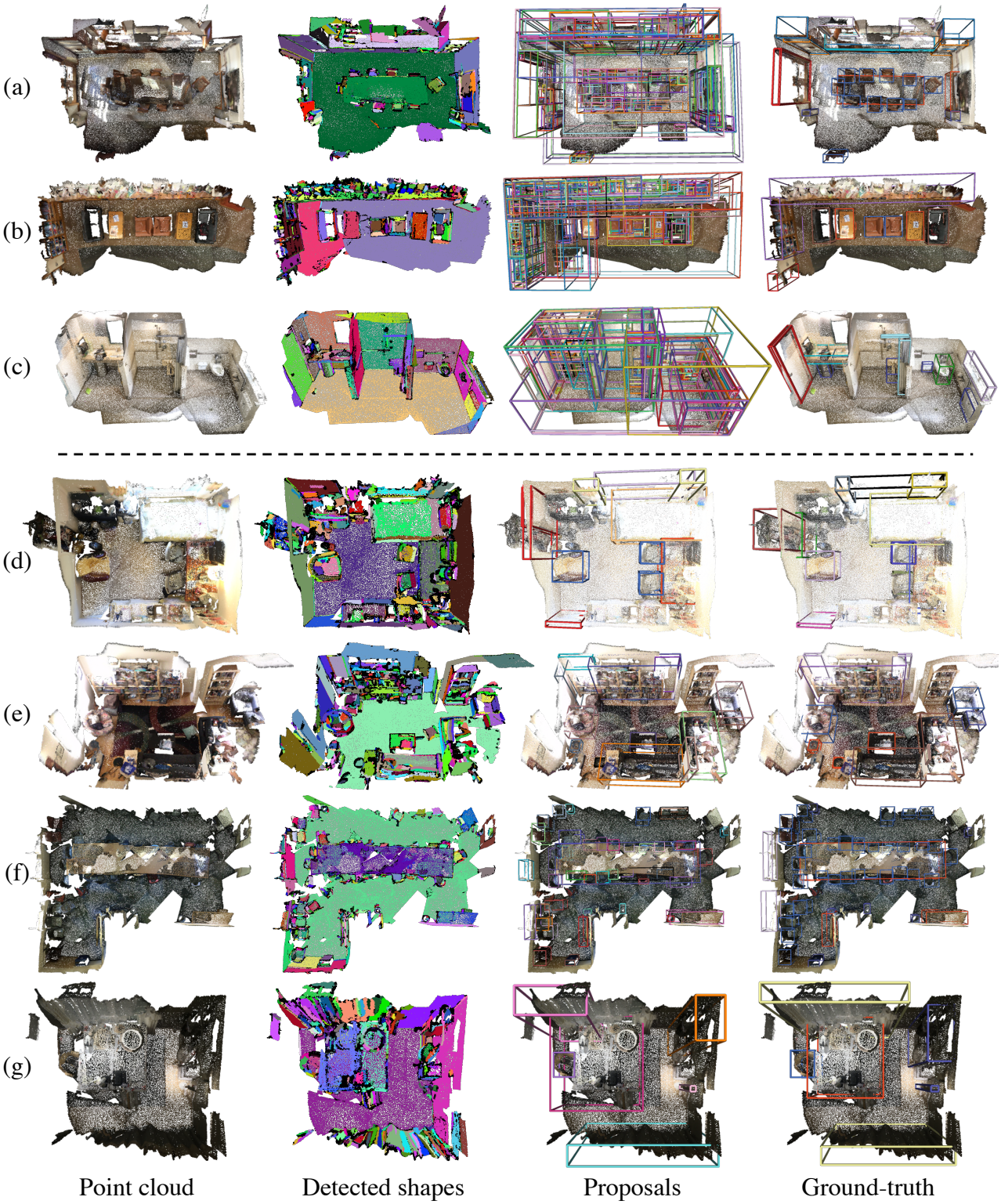

Figure 6: Visualization of the computed proposals. Top three rows show all the computed 3D proposals, from which we observe that the proposals are mainly around object areas. The bottom four rows show the proposals which best overlap with ground-truth boxes. GSS generates 3D proposals with great recall for various objects in complex scenes.

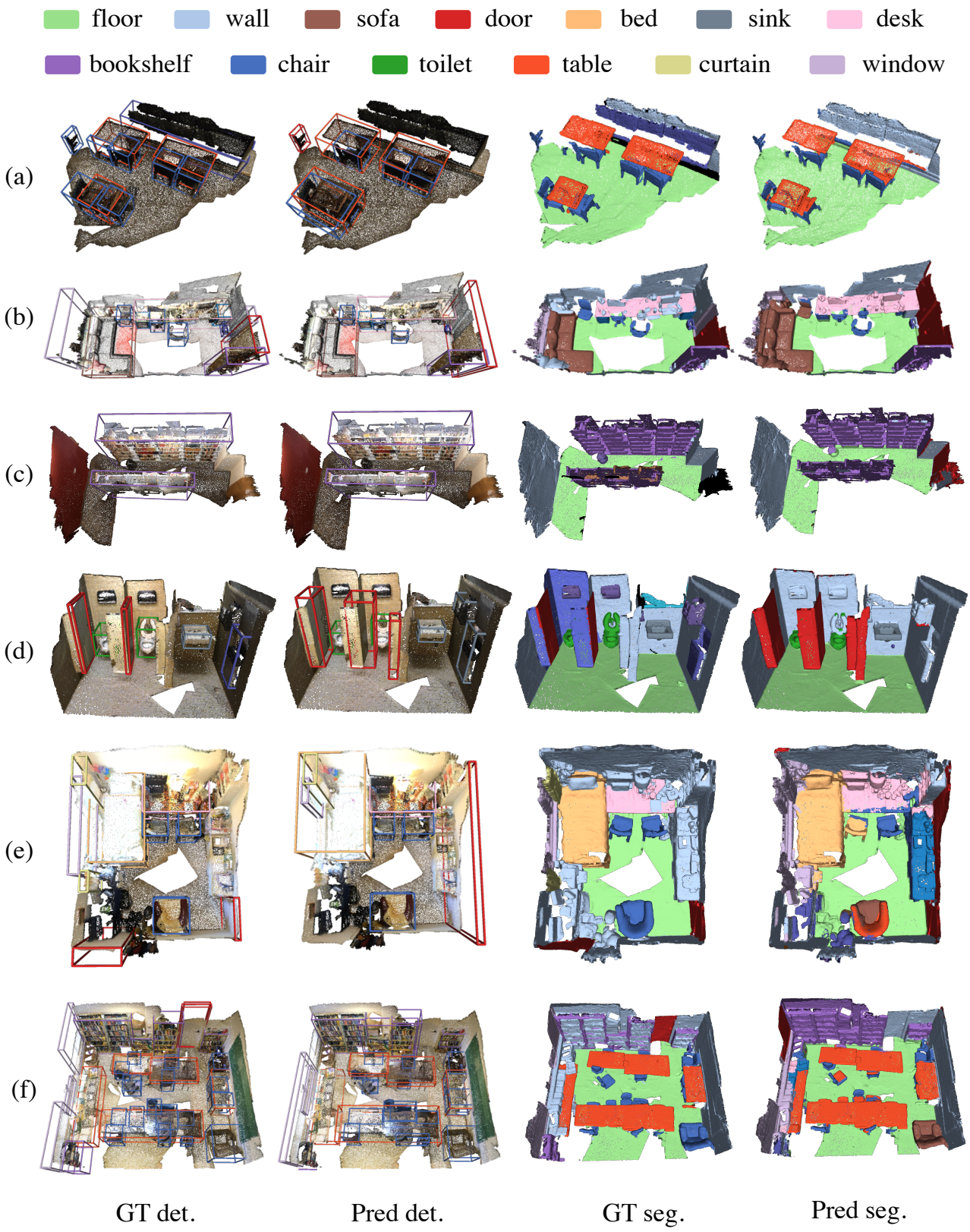

Figure 7: Additional qualitative results. We show the qualitative comparison between ground-truth labels and our (WyPR+prior) predictions. We show both detection and segmentation results for the same scene.