Prompt3D: Random Prompt Assisted Weakly-Supervised 3D Object Detection

Supplementary Material

Overview

This supplementary material is organized as follows:

- In Sec. A, the methodology for constructing our 3D shape library is detailed.
- Sec. B describes the specific process of virtual scene generation as discussed in the main paper.
- More experimental results are provided in Sec. C.
- Additional visualizations of the detection results are presented in Sec. D.
- Sec. F includes an analysis of parameter impacts, focusing on the loss weight.

A. 3D Shape Repository

Construction of Prompts. To create a 3D shape library, a wide variety of prompts for each category are produced using the large language model ChatGPT. Specifically, to generate prompts for a single category, the command (input prompt) provided to ChatGPT adheres to a consistent format:

"Please help me write N nouns about KEYWORD and optionally use clauses to describe the shape design of the KEYWORD. Each noun should not describe anything other than the shape, and start with a or an."

Here, N denotes the total number of generation prompts required, and *KEYWORD* specifies the target category, characterized by a singular term like "table". For instance, various prompts generated for the "bookshelf" category are showcased in Tab. S1. These randomly generated prompts cover a wide range of shapes pertaining to specific object types, thus ensuring considerable diversity. The prompts are converted into their respective shapes directly, bypassing any form of intermediate filtering or processing.

Construction of 3D Shapes. With the aid of Shap-E [2], these generated prompts are subsequently converted into 3D meshes. The architecture of Shap-E is divided into two phases: the first phase involves training a Transformer-based encoder to generate Implicit Neural Representation (INR) parameters for the 3D model, followed by training a diffusion model utilizing the encoder's output. Detailed specifics of this architecture are elaborated in [2]. As a result, an extensive and varied 3D shape library has been methodically established. This library comprises 22 classes, which are detailed in Fig. S1.

Keyword	*bookshelf*
Output	*An angular bookshelf with adjustable shelves*
	An angular bookshelf with adjustable shelves
	An asymmetrical bookshelf with asymmetrical compartments
	An arched bookshelf with curved shelves
	An asymmetrical bookshelf with staggered shelving units
	An artistic bookshelf with painted motifs
	An airy bookshelf with open back
	An adjustable bookshelf with customizable shelving options
	An angular bookshelf with geometric patterns
	An antique bookshelf with brass accents
	...

Table S1. Some examples of the generated prompts for the category "bookshelf".

B. Generation of Synthetic Scenes

Here we give a detailed construction process of the virtual scene.

Data Preparation. Before assembling the shapes into a virtual scene, we need to do some preparatory work.

Firstly, we calculate the height ($support\_z$) of the supporting surface for each generated shape. This involves identifying points where the dot product of the normal vector with $[0, 0, 1]$ exceeds 0.88, indicative of plane points. We then extract the z-axis coordinates of these points to form a one-dimensional array z_list, which is sorted in ascending order. Denoting the number of points as z_num, we select a subset $choose\_point = z\_list[\text{int}(z\_num \times 0.94) : \text{int}(z\_num \times 0.97)]$ and use $np.mean(choose\_point)$ to define $support\_z$. If the count of $choose\_point$ is less than 15, the range is adjusted to (0.90, 0.95). If it is less than 20, the range is modified to (0.85, 0.95). Should the count remain low, we use the range (0.80, 0.95), and if it is still insufficient, $\max(z\_list)$ is taken as $support\_z$.

Secondly, we compute the Minimum Enclosing Rectangle (MER) of each shape.

Thirdly, we assess whether a shape can function as a supporter. This is determined by extracting points whose distance from the height $support\_z$ is less than one-tenth of the object's height. We then calculate the surface area ($surface\_area$) of the MER formed by these points. Denoting the MER from the previous step as MER_area, if $surface\_area > MER\_area \times 0.9$, the shape is deemed a supporter, and the $Is\_supporter$ value is set to 0.

Additionally, for mesh-version shapes, we evaluate their watertightness. The majority of meshes produced by Shap-E are characterized by their watertight properties, contrasting with those from ModelNet, which frequently display

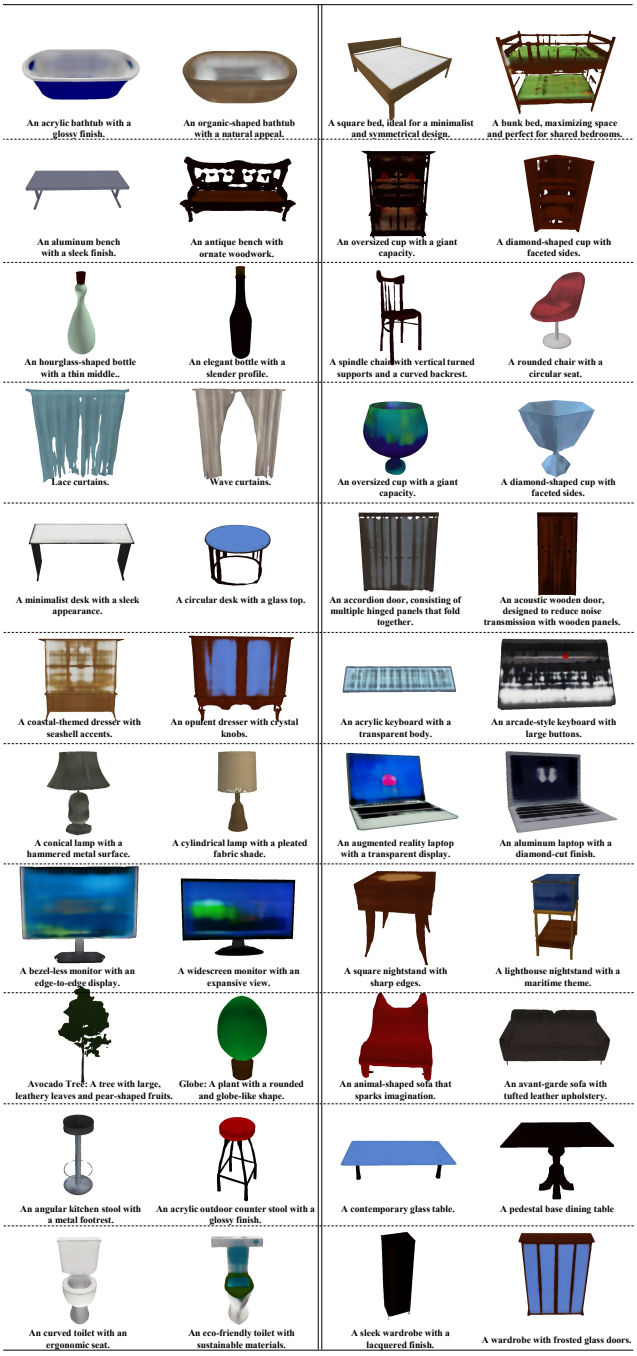


Figure S1. The visualized results of the reconstructed 3D shapes. Our 3D shape library includes 22 common categories in indoor scenes. Left column from top to bottom: bathtub, bench, bottle, cup, desk, dresser, lamp, monitor, plant, stool and toilet. Right column from top to bottom: bed, bookshelf, chair, curtain, door, keyboard, laptop, night stand, sofa, table, and wardrobe.

lower quality. This inferior quality is typically indicated by an abundance of boundary edges or non-manifold vertices.

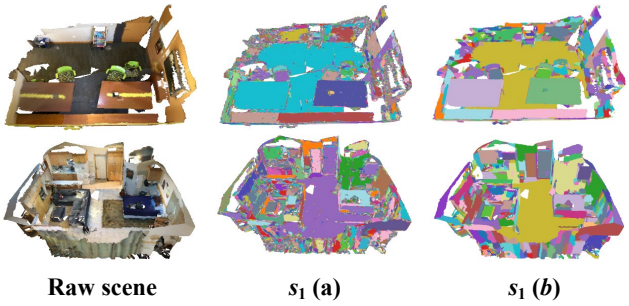


Figure S2. Details of the segmentation of the real scene. $\mathbf{s_1}$ $(\mathbf{a})$ involves segmenting the raw scene into multiple segments based on the mesh's geometric properties; $\mathbf{s_1}$ $(\mathbf{b})$ involves merging segments by combining smaller segments with adjacent segments.

Lastly, we extract weak annotation information from the real scene. The weak annotation for each object is represented as $[(x, y, z), modelnet\_id]$.

Segmentation of The Real Scene. The raw real scene in mesh format is aligned to the coordinate plane using the alignment matrix from the dataset, ultimately obtaining a point cloud $mesh\_vertices$ of dimension $n \times 3$. Points in $mesh\_vertices$ with a z-axis less than 0.05 are considered ground points, and their average value is taken as the ground height $ground\_z$; if there are no ground points, then $ground\_z = 0$. We segment the raw scene following the pipeline of [1], resulting in typical over-segmentation due to its unsupervised nature. As shown in Fig. S2, the segmentation process can be divided into 2 parts: s_1 (a) involves segmenting the raw scene into multiple segments based on the mesh's geometric properties; Step s_1 (b) involves merging segments by combining smaller segments with adjacent segments. Each color represents a segmented point set, *i.e.*, a segment (*seg*). Calculate the center of each *seg* along with its maximum and minimum values in the three coordinate dimensions. If the number of ground points in a *seg* exceeds 30 or half of the total, label it as -1, otherwise as 0 (to be later labeled as another object).

Initial Generation of The Virtual Scene. Here, we provide more details for each step from segmentation (Fig.3 $\mathbf{s_1}$) to the initial composition (Fig.3 $\mathbf{s_2}$).

We first collect the segs belonging to each object in the real scene (Fig.S3 $\mathbf{s_2}$ $(\mathbf{a})$). For each category, we obtain average 3D dimensions: dx_{avg}, dy_{avg}, dz_{avg}. Compute $radius_{avg} = (dx_{avg} + dy_{avg})/2$. In real-world scenarios, consider the centroid of an object as the focal point. Commence by extracting a point cloud within a sphere of radius $0.4 \times radius_{avg}$ centered on this point. Should the number of points within this sphere exceed 5, the selection

Detector	Setting	Method	batht.	bed	bench	bsf.	bot.	chair	cup	curt.	desk	door	dres.	keyb.	lamp	lapt.	monit.	n.s.	plant	sofa	stool	table	toil.	ward.	mAP@0.5
VoteNet	*full sup.*	FSB	69.8	76.9	6.7	26.0	0.0	67.6	0.0	10.2	30.0	13.3	21.1	0.0	15.5	0.0	19.6	47.9	3.1	70.4	10.1	38.9	85.0	2.7	28.0
	weak sup.	WSB	0.0	11.5	0.0	0.0	0.0	1.7	0.0	0.0	0.0	0.0	0.0	0.0	0.6	0.0	0.2	0.7	0.0	0.2	0.0	0.1	2.5	0.0	0.8
		WSBP	**37.1**	63.6	0.0	5.7	0.0	26.4	0.0	0.1	15.9	1.7	2.7	0.0	1.4	**0.1**	5.9	31.3	0.6	36.6	**11.3**	4.3	7.4	0.0	11.5
		BR_P	36.8	15.2	1.2	6.9	0.0	**42.7**	0.0	0.0	4.4	1.3	2.1	0.0	9.0	0.0	2.7	31.4	1.3	14.4	4.1	8.3	51.6	0.0	10.6
		BR_M	9.6	59.2	0.2	**12.8**	0.0	37.9	0.0	0.0	**22.1**	1.0	**6.2**	0.0	10.6	0.0	2.1	**44.6**	**2.7**	33.0	2.0	**25.3**	57.0	0.1	14.8
		Ours	4.0	**66.4**	**2.2**	10.1	0.0	35.2	0.0	**0.4**	20.2	**4.1**	5.2	0.0	**14.4**	0.0	**10.7**	23.9	2.3	**42.4**	6.9	20.0	**63.9**	**0.4**	**15.1**
GroupFree3D	*full sup.*	FSB	75.7	75.6	4.5	28.4	0.0	75.3	0.0	20.3	47.4	24.7	29.5	0.3	20.4	0.0	37.5	61.4	3.7	74.6	37.1	51.1	96.2	11.7	35.2
	weak sup.	WSB	1.9	24.7	0.0	0.1	0.0	31.2	0.0	0.0	0.1	0.1	0.0	0.0	6.5	0.0	2.1	1.5	0.1	2.6	2.0	0.5	54.3	0.0	5.8
		WSBP	51.4	41.3	0.0	0.5	**2.4**	41.2	0.0	0.4	2.1	0.3	1.7	0.0	23.5	4.2	14.5	38.4	0.4	6.5	17.0	0.9	71.0	0.0	14.4
		BR_P	**83.6**	**79.1**	0.0	10.8	0.0	53.5	0.0	0.0	0.0	1.6	3.7	0.0	19.6	**50.0**	6.5	60.0	16.7	21.1	5.7	14.6	90.1	0.0	23.5
		BR_M	83.3	65.0	0.0	4.1	0.0	**56.2**	0.0	0.5	11.8	2.1	16.7	**1.2**	**23.8**	12.5	16.0	**80.0**	**17.5**	42.2	28.6	28.0	99.2	0.0	26.8
		Ours	71.4	72.6	**0.1**	**12.1**	0.0	52.9	0.0	**4.8**	**32.5**	**6.3**	**22.8**	0.0	20.6	3.1	**32.9**	57.0	11.6	**71.3**	**32.5**	**30.5**	**100.0**	**21.5**	**29.8**

Table S2. 3D object detection results on ScanNet validation set, evaluated using mAP@0.5. Experiments were conducted with VoteNet and GroupFree3D detectors. Following Xu et al. [3], we use 22 categories to assess performance on ScanNet. Best scores in **bold**.

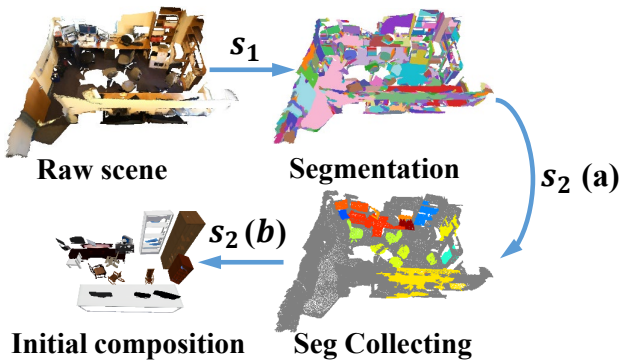


Figure S3. Details of the initial generation of the virtual scene. $\mathbf{s_2}$ $(\mathbf{a})$: collecting relevant segs for each real object; $\mathbf{s_2}$ $(\mathbf{b})$: loading the closest virtual object based on MER of each real object.

is deemed satisfactory. Otherwise, incrementally enhance the radius ratio from the initial value of 0.4 to 0.5, 0.6, and 0.7, respectively, for subsequent trials. In the event that these iterative attempts result in a null set, redefine the radius based on the larger of the two averages, $max_{avg} = \max(dx_{avg}, dy_{avg})$, and reassess the point count within a distance less than $prop \times max_{avg}$ from the object's centroid. Initiate this process with $prop$ set to 0.4 and incrementally increase it by 0.1 until the acquired point count surpasses 4. Now, for each object in real scenes, we find all points within a certain radius of its center, then identify the *segs* these points belong to. Should the z-axis span of a *seg* exceed $1.5 \times \max(dx_{avg}, dy_{avg})$, or if the z-coordinate of its center is below 0.15 or less than $0.2 \times dz_{avg}$, it will be excluded. Additionally, exclusion is warranted if the maximum distance between the object and seg centers in the x and y dimensions surpasses $1.5 \times \max(dx_{avg}, dy_{avg})$, or if the minimum distance is greater than $3.5 \times \min(dx_{avg}, dy_{avg})$. The same applies if the difference in the z-axis span is more than $1.5 \times dz_{avg}$. In cases where a *seg* is associated with two objects, it will be assigned to the one that is closer.

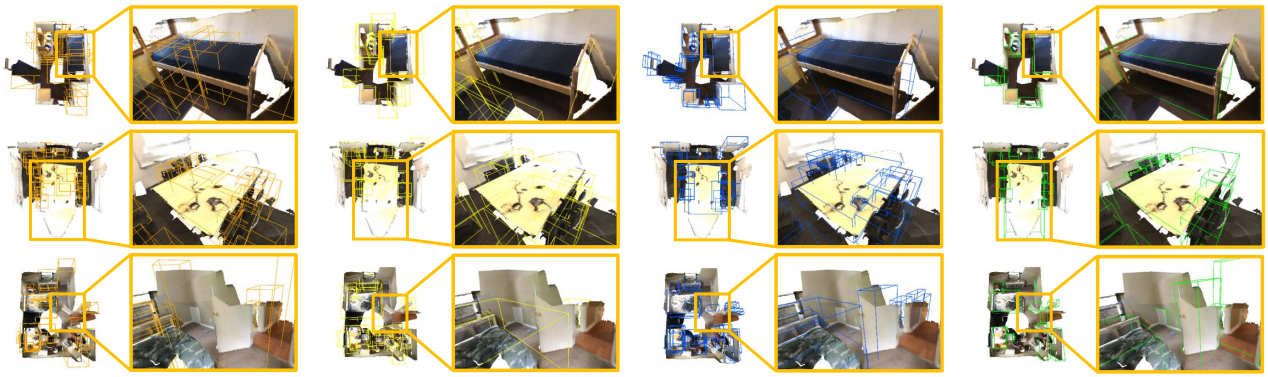
Then, we select the shape from our 3D shape repository that best matches the object in the real scene (Fig.S3 $\mathbf{s_2}$ $(\mathbf{b})$). After identifying non-intersecting *segs* for each object, extract all related point clouds based on these *segs*. Project these point clouds on the xy-plane to calculate the MER for the real object, represented as (l, s, θ), where l is the length, s is the width, and θ is the rotation angle. If l is greater than $5 \times max_{avg}$ or less than $0.25 \times min_{avg}$, we reset (l, s) to (max_{avg}, min_{avg}). Calculate the aspect ratio $ls\_ratio = l/s$. Find the 3d shape from the library with the closest ls_ratio.

At last, we replace the real object with the selected shape. The 3D mesh has already been preprocessed with MER data $(l, s, \theta, is\_supporter, support\_z)$. Now we determine the span of the three coordinate axes, denoted as ddx, ddy, and ddz, for a given shape, with pc_mean representing the centroid of this shape. Subsequently, we apply a transformation to the point cloud to reposition its centroid at the origin of the coordinate system. For each object in real scenes, calculate the set $[(x, y, z), scale, obj\_path, is\_supporter, \theta, MER, supporter\_z]$.

Here, (x, y, z) is the object center. supporter_list = ["desk", "bed", "bookshelf", "night_stand", "table"].

$is\_supporter$ is true if the object type is in the supporter list, otherwise false (in which case $support\_z$ is invalid). obj_path is the path of the shape, and MER belongs to the real object. Compute $\Delta\theta$ as the difference between the θ of the real object's point cloud and that of the virtual object. Rotate the 3D shape by an angle of $\Delta\theta$ and subsequently calculate the three-dimensional spans post-rotation, denoted as ddx, ddy, and ddz. Let the three-dimensional spans of the shape's point cloud be dx, dy, and dz.

For a generic category of the real object, define $scale\_z = \frac{(z - ground\_z) \times 2}{ddz}$. If the category is either table or chair and the minimum height of the real object is greater than 0.15, then $scale\_z = \frac{z_{max}}{ddz}$. If the category belongs to one of ["monitor", "plant", "lamp", "cup", "keyboard", "bottle", "laptop"], then $scale\_z = \frac{(z_{max} - z) \times 2}{ddz}$. Then, we can define $scale = (\frac{dx}{ddx}, \frac{dy}{ddy}, scale\_z)$. If the ob-



(a) WSB (b) BR (c) Ours (d) Ground Truth

Figure S4. Additional visualization of detection results, with zoomed-in details on the right for clarity. Comparing (a) WSB, (b) BR, (c) Our Prompt3D, and (d) ground truth labels on the ScanNet validation dataset.

ject category is among ["door", "curtain"], then $scale = (1, 1, scale\_z)$; if the category is "keyboard", then $scale = (\frac{dx}{ddx}, \frac{dy}{ddy}, \max(scale\_z, \frac{1.5 \times dx \times dy}{ddx * ddy}))$. $supporter\_z = z + scale\_z \times (support\_z - pc\_mean[2])$. Now we get the required information for each object in the synthetic scenes.

Adjustment of The Virtual Scene Layout. First, we conduct the gravity constraint. In the previous step, each object in the synthetic scenes is represented by $[(x, y, z), scale, obj\_path, is\_supporter, \theta, MER, supporter\_z]$. We also achieve the value of $ground\_z$. The gravity constraint involves adjusting each synthetic object's z. Let pc_txt represent the expansion of the number of point clouds of a real object into a one-dimensional vector across three dimensions. Then, process all the supporters. Adjust $new\_z = ground\_z - scale\_z * min(pc\_txt[:, 2])$. Next, process supportees. For non-supporters, check if their center point coordinates fall within the MER of a supporter. Find the corresponding supporter, and only search for supporters for these objects: ["monitor", "plant", "lamp", "cup", "keyboard", "bottle", "laptop"]. For those not found or not to be searched for, use the operation for supporters to place them on the ground (for "lamp", no action is taken). Otherwise, place them on the supporter's supporting surface: $new\_z = new\_positions[choosed\_supporter][6] - sz * min(pc\_txt[:, 2])$. we also obtained a $stage\_map$, with unsupported objects as keys, and the value is a list of objects placed on the key (which may be empty).

Secondly, we conduct the collision constraint. First, find the horizontal center (x, y) of the $floor\_points$ on the ground. For all unsupported objects, sort them by their distance from the center of the ground. Process each object in turn. If an object overlaps with some previous objects, calculate the direction of movement. If the distance to an object is (dx, dy), then sum up $(\frac{1}{dx}, \frac{1}{dy})$ for all colliding objects to find the direction of movement. Move the object in this direction until it no longer collides. For supported objects, follow the movement of the object supporting it. Then, sort them by the distance from the center of the supporting object. Process each object in turn. If an object overlaps, calculate the direction of movement and move it accordingly until the collision is resolved.

C. Detection Results of mAP@0.5

We assessed the efficacy of our weakly supervised method using the mAP@0.5 metric. The quantitative findings are delineated in Tab. S2. On the ScanNet validation set, the trend in mAP@0.5 results echoes that of mAP@0.25. Notably, our approach outshines all competing methods listed in Tab. S2. When VoteNet is employed as the detector, our method shows a 14.3% increase in mAP@0.5 compared to WSB, evidencing its capability to reduce information loss in the weakly supervised setting. Additionally, our technique exceeds the performance of the BR approach, which is dependent on synthetic scenes created from ModelNet40. When utilizing GroupFree3D as the detector, our approach demonstrates even more pronounced superiority under the weakly supervised setting. Our method registers a 24.0% improvement in mAP@0.5 compared with the WSB method and significantly surpasses the BR approach by 3% points.

Moreover, pre-training two distinct object detection networks with our synthesized scenes has proven to enhance their performance in weakly supervised settings. This improvement is likely due to the richer 3D shapes and the additional prior information provided by our synthetic scenes for the pre-training phase of object detection models.

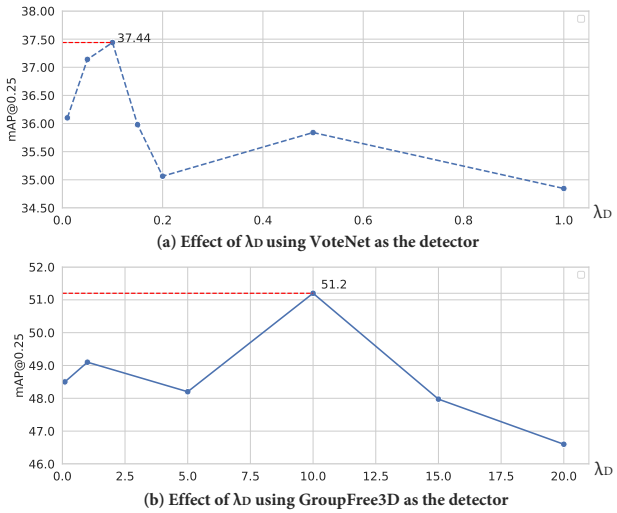


Figure S5. The mAP@0.25 results w.r.t. λ_D. When utilizing VoteNet as the detector, our method attains its peak performance when λ_D is set to 0.1. On the other hand, when employing GroupFree3D as the detector, setting λ_D to 1 yields the optimal performance.

Figure S6. Synthetic Scene built under various settings of θ.

D. Additional Visualization Results

In Fig. S4, we provide additional visualization of the detection results of WSB, BR, and our method using VoteNet as the detector. Our method consistently succeeds in scenarios where WSB and BR falter. These results further affirm the benefits of label augmentation through virtual scenes generated from prompts, demonstrating the effectiveness of our approach in improving detection accuracy.

Factor	Setting	mAP@0.25
z	0.01	36.2
	0.05	**37.4**
	0.1	34.0
θ	0	34.4
	Random θ	34.8
	$\Delta\theta$	**37.4**

Table S3. Ablation study on SSG.

E. More ablation studies of SSG

In the main paper, we focus on testing the key hyperparameters/modules affecting performance. Here we show more details about the settings of the SSG module.

Hyperparameters of SSG We carefully differentiate between parameters of the SSG module with physical significance and those offering adaptability for diverse datasets. For instance, in real scenes, points with a z-axis value z less than 0.05m are considered ground points. We kindly provided additional experiments regarding z in Tab. S3. Other parameters cover a wide search range, are determined by the inner parameter of the dataset. For instance, in Sec B, we gradually expand the search radius to collect all $segs$ belonging to real objects. In short, SSG hyperparameters are fixed and versatile.

Heading Consistency in SSG For orientation alignment, the heading information is not required. Orientation is refined through the calculation of angular difference ($\Delta\theta$) between the MER of real and synthetic objects, ensuring a harmonious alignment between real and synthetic entities. We conducted ablation experiments about the rotation angle θ to validate the effectiveness of our method (shown in Fig. S6 and Tab. S3).

F. Loss weight

The loss function for training the network mentioned in the main text is:

$$\mathcal{L}_{total} = \mathcal{L}_R + \lambda_S \mathcal{L}_S + \lambda_D \mathcal{L}_D, \quad (1)$$

Following the prior work [3], we set $\lambda_S = 0.1$. To ascertain the optimal value for the loss weight λ_D, we conducted tests with various values while maintaining all other hyperparameters constant.

As depicted in Fig. S5, optimal performance is attained when λ_D is set to 0.1 with VoteNet as the detector. Conversely, when employing GroupFree3D as the detector, the model reaches its peak efficacy at $\lambda_D = 1$. Consequently, we have determined the values of various coefficients in the loss function.

References

[1] Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient graph-based image segmentation. *IJCV*, 59:167–181, 2004. 2

[2] Heewoo Jun and Alex Nichol. Shap-e: Generating conditional 3d implicit functions. *arXiv preprint arXiv:2305.02463*, 2023. 1

[3] Xiuwei Xu, Yifan Wang, Yu Zheng, Yongming Rao, Jie Zhou, and Jiwen Lu. Back to reality: Weakly-supervised 3d object detection with shape-guided label enhancement. In *CVPR*, pages 8438–8447, 2022. 3, 5