A Experiment Details

This section provides more implementation details of the proposed Object2Scene approach and L3Det model.

Prompt Generation for Object2Scene Here we provide detailed information for the grounding prompt generation process introduced in Section 3.2. Following SR3D, we generate the spatial prompt for the inserted target object using the following template: $\langle target - class\rangle\langle spatial - relationship\rangle\langle anchor - class\rangle$. Since the target and anchor classes are determined during 3D object insertion, we need to describe the spatial relationship according to their relative locations. we divide the spatial object-to-object relations into three categories:

1. **Vertical Proximity:** It indicates the target is *on* the anchor object.
2. **Horizontal Proximity:** This indicates the target is around the anchor object and can be represented by words like: *next to* or *close to*.
3. **Allocentric:** Allocentric relations are actually based on Horizontal Proximity, which encodes information about the location of the target with respect to the self-orientation of the anchor, which can be represented by words like: *left*, *right*, *front*, *back*.

Once we obtain the generated spatial prompt such as *"the table that is next to the bar stool."*, given a text sentence of anchor object *"it is a wood bar stool. The stool is in the kitchen at the bar. It is the very first stool at the bar."* in ScanRefer (Chen et al., 2020), we utilize the off-the-shelf tool to decouple the text and obtain the main object, auxiliary object, attributes, pronoun, and relationship of the sentence as shown in Figure 5, following EDA (Wu et al., 2022). Then we replace the main object in the sentence with the inserted target object and the original main object becomes the auxiliary object. Combining the spatial prompt, the generated grounding prompt could be *"it is a table next to a wood bar stool. The stool is in the kitchen at the bar. The stool is the very first stool at the bar."*

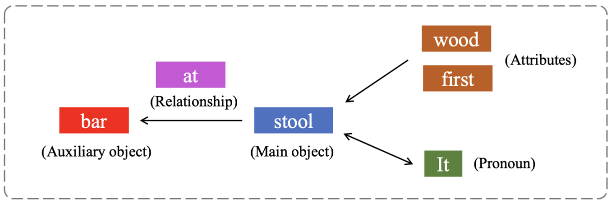

Figure 5: Sentence decoupling illustration.

Details of the model architecture of L3Det In our proposed model L3Det, the point cloud feature tokens $\mathbf{V} \in \mathbb{R}^{n\times d}$ are extracted by PointNet++ (Qi et al., 2017) encoder pre-trained on ScanNet (Dai et al., 2017) seen classes, where $n = 1024$ denotes the number of input points. The text query tokens $\mathbf{T} \in \mathbb{R}^{l\times d}$ are extracted by the pre-trained RoBERTa (Liu et al., 2019) text encoder, where $l = 256$ is the maximum length of the text. The non-parametric queries are predicted with an MLP from the 256 visual tokens with the highest scores. Besides, the number of layers in the decoder is set to $N_E = 6$. The decoder predicts object features $\mathbf{O} \in \mathbb{R}^{k\times d}$, where $k = 256$ is the number of candidate objects, and $d = 288$ is the feature dimension.

Performance in the process of BUTD-DETR simplification to L3Det Table 7 shows the changes in the visual grounding performance from top to bottom during the process of simplifying BUTD-DETR to our L3Det. From the results, it can be seen that by directly inputting text tokens and object queries parallelly into the decoder, it can compensate for the performance drop caused by abandoning the cross-encoder and even achieve better performance (51.0 $\rightarrow$ 47.2 $\rightarrow$ 51.3). Besides, using alignment loss following GLIP can further improve the model's performance to 52.8 ($>$ 52.2, the performance of BUTD-DETR) while not using box stream compared with BUTD-DETR.

Comparision with existing detection methods for L3Det To demonstrate our L3Det's strong detection capability, we directly train L3Det on ScanNet 18 classes using the 18 categories combination detection prompt, and the experiments in Table 6 show L3Det achieves higher detection performance.

Table 5: Performance change in the architecture modification from BUTD-DETR to L3Det.

Method	**Accuracy**
BUTD-DETR	52.2
+ remove box stream	51.0
+ with concatenated Visual and Language Streams	50.1
+ remove cross-encoder	47.2
+ replace with our L3Det decoder (using parallel text and object query	51.3
+ replace with GLIP alignment loss (Our L3Det)	52.8

Table 6: 18 class 3D object detection results on ScanNetV2.

Method	mAP_{50}
3DETR	44.6
GroupFree3D	48.9
L3Det	50.1

Combining Object2Scene with existing methods Here we attempt to use Object2Scene to enable the close-set 3D object detector to obtain open-vocabulary detection capability. Since both GroupFree3D and 3DETR are close-set 3D object detectors and do not possess the text input capability, we modified their class prediction to 20 classes, which cover all the categories in OV-ScanNet20 benchmark, but only seen annotations (10 classes) are used for training in the actual training process. Then we introduce unseen objects using Object2Scene to expand the training dataset. Results on OV-ScanNet20 in Table 7 show our Object2Scene is general, and L3Det is also better than GroupFree3D and 3DETR due to the text prompt input ability and architecture advantages.

Table 7: Detection results on unseen classes of OV-ScanNet20.

Method	mAP_{25}
3DETR	1.31
GroupFree3D	0.53
3DETR + Object2Scene	14.23
GroupFree3D + Object2Scene	15.16
L3Det + Object2Scene	23.98

Alignment Matrix Generation for L3Det In the second paragraph of Section 4.1 of the main paper, we introduce the training supervision for L3Det, where calculating the alignment loss requires a target alignment score matrix $\mathbf{S}_{target} \in \{0, 1\}^{N \times M}$. The key to generating the target alignment score matrix is the fine-level alignment between the text tokens and 3D boxes which is typically not provided in most of visual grounding datasets including ScanRefer (Chen et al., 2020). We use the off-the-shelf tool following EDA (Wu et al., 2022) to parse the text description, generate the grammatical dependency trees, and obtain the position label. For example, given a sentence *"It is a white table. It is next to a backboard"* consisting of multiple objects, the main object in this sentence is *"table"* and the corresponding position label is *"0000100...."*.

Training Details The code is implemented based on PyTorch. Our model is trained on two NVIDIA A100 GPUs with a batch size of 24. We freeze the pretrained text encoder and use a learning rate of 1×10^{-3} for the visual encoder and a learning rate of 1×10^{-4} for all other layers in the network. It takes around 25 minutes to train an epoch, and our model is trained for 120 epochs. The best model is selected based on the performance of the validation set.

B Visual Grounding Results

Our proposed L3Det model unifies the 3D object grounding and detection with the same framework, and we report the language-based 3D grounding performance trained on ScanRefer (Chen et al., 2020) in Table 8. Compared with previous works such as BUTD-DETR (Jain et al., 2022), our proposed L3Det achieves better grounding results with a simpler model architecture. We hope our proposed L3Det will serve as a new 3D grounding and detection baseline for its simple, effective, and unified model architecture.

Table 8: Performance comparisons on language grounding on ScanRefer (Chen et al., 2020)

Method	Unique@0.25	Unique@0.5	Multi@0.25	Multi@0.5	Overall@0.25	Overall@0.5
ReferIt3DNet (Achlioptas et al., 2020)	53.8	37.5	21.0	12.8	26.4	16.9
ScanRefer (Chen et al., 2020)	63.0	40.0	28.9	18.2	35.5	22.4
TGNN (Huang et al., 2021)	68.6	56.8	29.8	23.2	37.4	29.7
IntanceRefer (Yuan et al., 2021)	77.5	66.8	31.3	24.8	40.2	32.9
FFL-3DOG (Feng et al., 2021)	78.8	67.9	35.2	25.7	41.3	34.0
3DVG-Transformer (Zhao et al., 2021)	77.2	58.5	38.4	28.7	45.9	34.5
SAT-2D (Yang et al., 2021)	-	-	-	-	44.5	30.1
BUTD-DETR (Jain et al., 2022)	84.2	66.3	46.6	35.1	52.2	39.8
L3Det	**84.8**	**67.1**	**47.1**	**35.9**	**52.8**	**40.2**

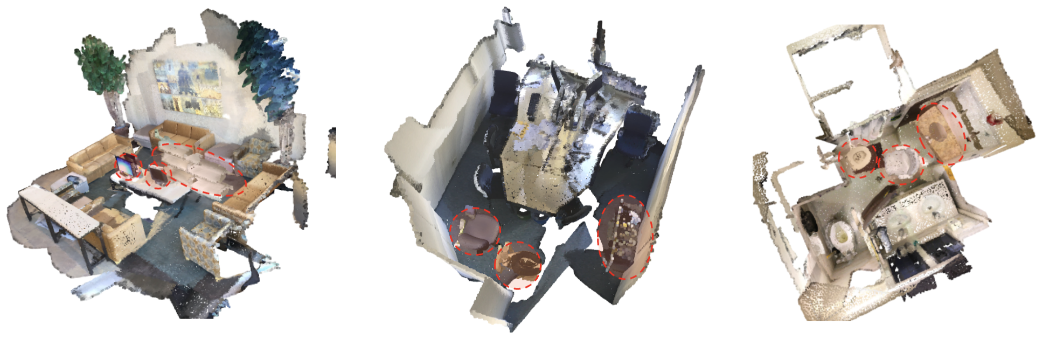
Figure 6: Sample scenes generated by Object2Scene. The objects surrounded by the red circle in the figure are sampled from 3D object datasets and inserted into the real-scanned scene.

C Qualitative Analysis

In this section, we illustrate more scenes generated by our Object2Scene approach in Figure 6 and more visualization results in Figure 7. Figure 6 shows several scenes generated by Object2Scene, where the 3D objects are inserted into the real-scanned scenes in a reasonable manner. As illustrated in Figure 7, L3Det can locate all objects belonging to the category described in the input text prompt covering various object sizes. Nevertheless, we find that our model may sometimes incorrectly detect the objects (illustrated in the top middle sub-figure in Figure 7) or miss the objects. For example, if the chairs are tucked under the table, the actual point cloud distribution of the chairs and the point cloud distribution of chairs we insert into by Object2Scene are often very different, making it difficult to detect. Those failure cases might be due to the distribution misalignment between the scanned point cloud in the scene and the point cloud of the inserted objects from other datasets. We leave this issue for future work.

Table 9: Ablation Study.

(a) Performance of different training epochs when 40% of the 3D objects from the 3D object dataset are used for training.

Training Epochs	mAP_{25}
30	11.87
45	15.43
60	18.99
100	20.62
120	21.31

(b) Performance of different data ratio used in Object2Scene with 120 training epochs. Data ratio refers to the ratio of objects from the 3D object dataset that are used for training. It reflects the diversity of 3D objects that are inserted to the scenes.

Data Ratio	mAP_{25}
40%	12.56
80%	18.11
100%	21.31

D Ablation Study

In this section, we provide more ablation studies on how to use the data generated by Object2Scene for training. During training, the Object2Scene approach generates augmented scenes online, *i.e.*, the inserted objects and locations to insert objects are sampled at each iteration. We investigate two factors: 1) the number of training epochs, and 2) the diversity of inserted objects, *i.e.*. the ratio of

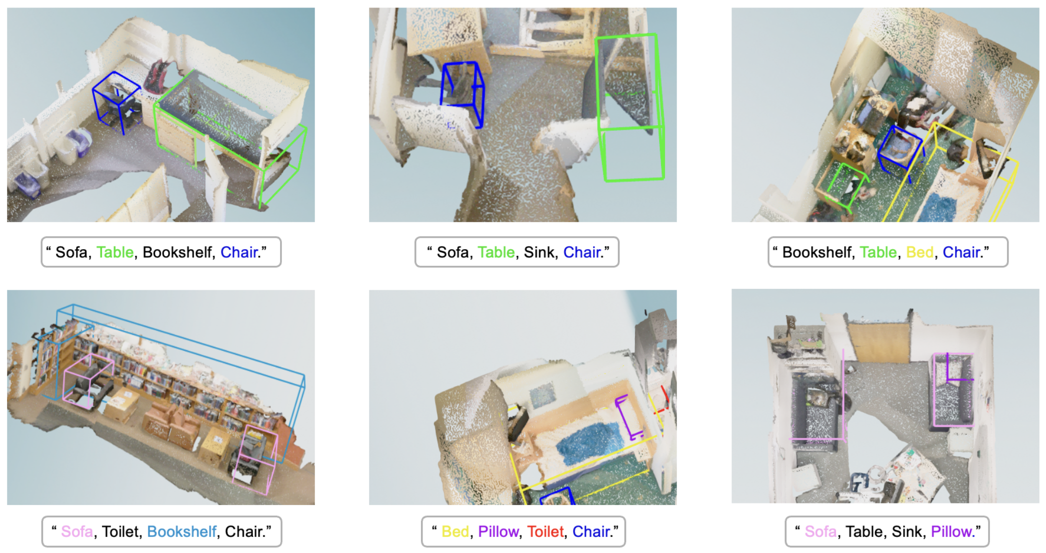
“ Sofa, Table, Bookshelf, Chair.”
“ Sofa, Table, Sink, Chair.”
“ Bookshelf, Table, Bed, Chair.”
“ Sofa, Toilet, Bookshelf, Chair.”
“ Bed, Pillow, Toilet, Chair.”
“ Sofa, Table, Sink, Pillow.”

Figure 7: Qualitative results for open-vocabulary 3D object detection results. For each scene, the detection prompt is shown under the input point cloud. The colors of bounding boxes correspond to the classes in the prompts.

data from the 3D object dataset that is used for training. Table 9a demonstrates that more training epochs lead to better performance. Table 9b indicates that increasing the diversity of inserted 3D objects improves the performance.

E Transferability to New Datasets

We explore the transferability of our 3D detectors by evaluating the cross-dataset transfer performances between OV-ScanNet20 and OV-SUN RGB-D20. The transferability of our 3D detector mainly comes from the pretrained text encoder and the robust and transferable 3D feature representations trained with objects from multiple source datasets using the cross-domain category-level contrastive loss. The object detector trained on OV-ScanNet20 achieves an mAP_{25} of 16.34% when tested on OV-SUN RGB-D20 dataset, and the object detector trained on OV-SUN RGB-D20 achives an mAP_{25} of 17.11% when tested on OV-ScanNet20, demonstrating the transferability of the object detectors trained with Object2Scene.