Supplementary for NS3D: Neuro-Symbolic Grounding of 3D Objects and Relations

The appendix is organized as the following. In Appendix A, we formally define the domain-specific language (DSL) used by NS3D. In Appendix B, we provide dataset details, additional qualitative examples on both 3D-REC and 3D-QA tasks, and an additional experiment on scene complexity generalization, where we train models on scenes with a small number of objects but test on larger and more complex scenes.

A. Domain-Specific Language

In this section, we summarize the value types (Table 1) and function definitions (Table 2) of the domain-specific language used in our paper. The *scene* operation is parameter-free, while other functions take input *object_set*'s and output an *object_set*, represented as the object score vector. Query-type operations take input *object_set*'s and output answers of the target type, such as Boolean values (*e.g.*, "is there a chair?") and concept names (*e.g.*, "what is the type of the object next to the table?"). Existence and counting-related operations involve a score threshold $t = 0.8$, which is a scalar hyperparameter. In our experiment, the threshold t is chosen over a separate 3D-QA dataset based on scenes from the train set, instead of the test set. The *query_object*, *query_relation*, and *query_t_relation* operations are implemented through finding the object or relation label based on object score vectors.

Zero-shot transfer to 3D-QA. NS3D composes learned models on 3D-REC to build new 3D-QA operators in a zero-shot manner, requiring no additional training. The 3D-QA modules can be implemented by reusing the MLPs learned for object and relation classification from the 3D-REC task. Intuitively, let us consider *query_object* in 3D-QA, which takes an object as input and outputs its category. Since we have already learned classifiers for all categories (MLPs used in the *filter* operation), NS3D directly reuses these modules to answer the question: it evaluates all MLP classifiers on the object feature and returns the category with the highest score.

Type	Representation	Semantics
object_set	Object score vectors.	Set of objects selected from a scene.
category	Concept names: *table, chair, piano,* etc.	Object-level properties.
relation	Concept names: *near, left, behind,* etc.	(Binary) Relationships between two objects.
t_relation	Concept names: *between, anchor-left* etc.	(Ternary) Relationships among three objects.
**boolean*	Strings: *yes, no*.	Boolean values.
**integer*	Integers: *0, 1, 2,* etc.	Count of objects.

Table 1. Types in the NS3D domain-specific language. *: Types that are only used in the 3D-QA task.

Signature & Implementation	Semantics
scene() $\longrightarrow y$: *object_set* **Implementation:** $y_i = 0$, for all $i \in \{1, 2, \cdots, M\}$	Return all objects in the 3D scene.
filter(x: *object_set*, c: *category*) $\longrightarrow y$: *object_set* **Implementation:** $y_i = \min\left(x_i, prob_i^{\mathrm{c}}\right) = \min\left(x_i, \mathrm{MLP}^{\mathrm{c}}\left(f_i^{\mathrm{obj}}\right)\right)$	Return all objects satisfying a concept c.
relate(x^t: *object_set*, x^r: *object_set*, *rel*: *relation*) $\longrightarrow y$: *object_set* **Implementation:** $prob_{i,j}^{\mathrm{rel}} = \mathrm{MLP}^{\mathrm{rel}}(f_{i,j}^{\mathrm{rel}})$ $y_i = \min\left(x_i^t, \sum_j sx(x^r)_j \cdot prob_{i,j}^{\mathrm{rel}}\right)$	Return all objects that satisfy the relationship *rel* between the object sets.
ternary_relate(x^t: *object_set*, x^{r1}: *object_set*, x^{r2}: *object_set*, *trel*: *t_relation*) $\longrightarrow y$: *object_set* **Implementation:** $prob_{i,j,k}^{\mathrm{trel}} = \mathrm{MLP}^{\mathrm{trel}}(f_{i,j,k}^{\mathrm{ternary}})$ $y_i = \min\left(x_i^t, \sum_j \sum_k sx(x^{r1})_j \cdot sx(x^{r2})_k \cdot prob_{i,j,k}^{\mathrm{trel}}\right)$	Return all objects that satisfy the ternary relationship *trel* between the object sets.
anchor(*object_set*, *object_set*) $\longrightarrow$ *object_set* **Implementation:** internally handled using *ternary_relate*.	Return all objects that satisfy the relationship anchored on the first object set.
**query_exist*(x: *object_set*) $\longrightarrow y$: *boolean* **Implementation:** $y = \begin{cases} yes & \text{if } \max_i(\sigma(x_i)) > t \\ no & \text{if } \max_i(\sigma(x_i)) \leq t \end{cases}$	Return Yes/No corresponding to existence of object in the object set.
**query_count*(x: *object_set*) $\longrightarrow y$: *integer* **Implementation:** $y = \sum_i \mathbb{1}\left[\sigma(x_i) > t\right]$	Return count of objects in the object set.
**query_object*(x: *object_set*) $\longrightarrow c$: *category* **Implementation:** $c = \arg\max_{\mathrm{c}}\left(\sum_i sx(x)_i \cdot \mathrm{MLP}^{\mathrm{c}}\left(f_i^{\mathrm{obj}}\right)\right)$	Return type of object in the object set.
**query_relation*(x^t *object_set*, x^r: *object_set*) $\longrightarrow$ *rel*: *relation* **Implementation:** $rel = \arg\max_{\mathrm{rel}}\left(\sum_i \sum_j sx(x^t)_i \cdot sx(x^r)_j \cdot \mathrm{MLP}^{\mathrm{rel}}(f_{i,j}^{\mathrm{rel}})\right)$	Return relationship between the object sets.
**query_t_relation*(x^t:*object_set*, x^{r1}: *object_set*, x^{r2}: *object_set*) $\rightarrow$ *trel*: *t_relation* **Implementation:** $trel =$ $\arg\max_{\mathrm{trel}}\left(\sum_i \sum_j \sum_k sx(x^t)_i \cdot sx(x^{r1})_j \cdot sx(x^{r2})_k \cdot \mathrm{MLP}^{\mathrm{trel}}(f_{i,j,k}^{\mathrm{ternary}})\right)$	Return ternary relationship between the object sets.

Table 2. Primitive functions defined in the NS3D domain-specific language. *: Functions that are only used in the 3D-QA task. Here, $sx(\cdot)$ is the Softmax function, $\sigma(\cdot)$ is the Sigmoid function, and $\mathbb{1}[\cdot]$ is the indicator function which returns 1 when the expression inside the brackets evaluates to true, and 0 otherwise.

B. Experimental Details and Additional Results

In this section, we first present details for the datasets used in the main text. Then, we provide additional results on the scene complexity generalization task, where we train models on scenes with a small number of objects but test on larger and more complex scenes. Finally, we showcase additional qualitative examples for both the 3D-REC and 3D-QA tasks.

B.1. Dataset

ReferIt3D datasets. For settings where NS3D was trained on the full ReferIt3D dataset, we used the exact SR3D training data for all networks, including 707 scenes with object category annotations and $65,844$ query-answer pairs in total.

Data efficiency datasets. We generated data-efficient train sets with randomly sampled 0.5% (329 examples), 1.5% (987 examples), 2.5% (1,646 examples), 5% (3,292 examples), and 10% (6,584 examples) of the train set, with the same full test set from SR3D used for evaluation.

***PAIRS* and *SCENE* generalization datasets.** We created two new datasets to test generalization ability. Both of the datasets are built on the SR3D train and test set.

The first dataset (PAIRS) evaluates performance on unseen object-relation-object pairs. The referring expressions in the train set include the top 5 percent of object-relation-object pairs: *i.e.*, the referred object category, relation type, and the reference object category (*e.g.*, chair-closest-door). The test set contains the bottom 95 percent of object-relation-object pairs in the long-tailed distribution. The train set and test set consists of 16,200 examples and 10,520 examples respectively.

The second dataset (SCENE) evaluates performance on an unseen scene type. The train set includes train examples with all scene types aside from that of “living room”, while the test set only contains examples in living rooms. The train set and test set consists of 57,125 examples and 1,320 examples respectively.

3D-QA dataset. We manually created a small evaluation set of 50 examples for the 3D-QA task, based on the test set of ReferIt3D [1]. The input is a set of objects in the scene, $\mathcal{O} = \{O_1, ..., O_M\}$, and a question $\mathcal{Q}$. In contrast to the 3D-REC task, where the output is the target object, the output for 3D-QA is an answer in text form (the vocabulary contains all categories, relations, Yes/No, and integers). The dataset consists of four main types of questions created from the following templates:

Existence-typed questions:

- Is there a [`Object`] [`Relation`] [`Object`]? A: Yes/No
- Is there a [`Object`] [`Relation`] [`Object`] and [`Object`]? A: Yes/No
- Facing [`Object`], is there a [`Object`] [`Relation`] [`Object`]? A: Yes/No

Counting-typed questions:

- How many [`Object`] are in the scene? A: Integer
- How many [`Object`] are [`Relation`] [`Object`]? A: Integer

Object-typed questions:

- What is the item [`Relation`] [`Object`]? A: [`Object`]
- What is the item [`Relation`] [`Object`] and [`Object`]? A: [`Object`]
- Facing [`Object`], what is the item [`Relation`] [`Object`]? A: [`Object`]

Relation-typed questions:

- What is the relationship between [`Object`] and [`Object`]? A: [`Relation`]
- Facing [`Object`], what is the relationship between [`Object`] and [`Object`]? A: [`Relation`]

B.2. Scene Complexity Generalization

For all experiment results reported in the main text, NS3D was trained on examples with only 10 objects given in the scene, and evaluated on the full test set with up to 88 objects in the scene. NS3D is able to show this scene complexity generalization, as it does not need the full scene point cloud as its input and instead only explicitly models a given object set and relations between specified objects. This improves training efficiency, reduces the need for annotated 3D objects, which are expensive to acquire in 3D domains, and enables generalization to more cluttered scenes.

	OVERALL	VIEW-DEP.
NS3D (OURS)	**0.627**	**0.620**
MVT [3]	0.405	0.396
SAT [5]	0.444	0.415
TRANSREFER [2]	0.360	0.344

Table 3. Generalization results from sparse scenes to dense scenes.

We show that baselines methods cannot generalize as NS3D does, and yields significantly decreased performance when trained on 10 objects per scene and evaluated on more complex scenes. In Table 3, we see that NS3D outperforms prior works by a large margin in this setting. We did not test BUTD-DETR, because BUTD-DETR explicitly encodes the full 3D scene as input, with all objects given in train and test, and hence does not directly apply to this partial scene setup.

B.3. NR3D Results

We report results on NR3D, the natural language variant of ReferIt3D. While NS3D does work on natural language, as Codex can parse NR3D input into programs, Codex parsing yields 91 distinct function modules and 5892 concepts, resulting in a separate long-tailed problem. Hence, we ran additional experiments on a subset of NR3D, by restricting utterances to those that parse to the same set of functions and concepts in SR3D, which yields 3659 train examples and 1041 test examples.

	OVERALL	VIEW-DEP.
NS3D (OURS)	**0.526**	**0.432**
BUTD-DETR [4]	0.382	0.331
MVT [3]	0.430	0.324
SAT [5]	0.329	0.248
TRANSREFER [2]	0.360	0.286

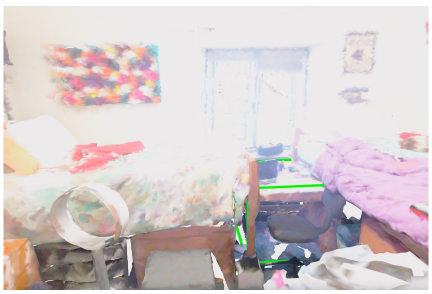
Table 4. Results on a constrained version of NR3D from the ReferIt3D datasets.

We train NS3D as well as top-performing baselines, and see that NS3D significantly outperforms prior work (Table 4). This suggests that NS3D can learn from natural language data in a data-efficient way. Examples from this NR3D subset include noisy natural language such as "The picture above the bed with the laptop on it." and "The monitor that you would class as in the middle of the other two"; both exhibit noisy natural language, with more complex underlying programs than the SR3D training set.

B.4. Qualitative Examples

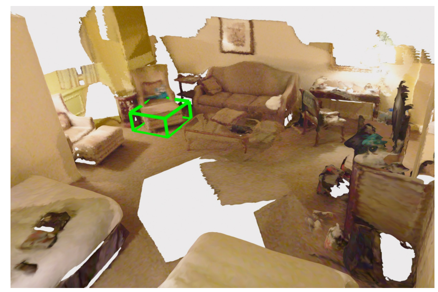
In Figure 1, we show additional examples of the ReferIt3D 3D-REC task in SR3D, with examples of binary and ternary relations. In Figure 2, we present comparisons of NS3D against baselines on view-dependent examples, with the green outline indicating correct selection and red outline indicating incorrect selection. We see that NS3D is able to outperform prior work in disambiguating the target referred object.

In Figure 3, we present additional qualitative examples of NS3D on the 3D-QA task. We see examples of success cases in green and failure cases in red for NS3D in the zero-shot transfer setting.



Instruction: Select the **desk** that is supporting the laptop.

Instruction: Looking at the front of the bed, select the **nightstand** that is on the right side of it.



Instruction: Select the **ottoman** that is in the center of the desk and the end table.

Instruction: Select the **microwave** that is close to the kitchen cabinets.

Instruction: Facing the front of the couch, choose the **towel** that is on the left side of it.

Instruction: Choose the **suitcase** that is in the middle of the bathtub and the cabinet.

Figure 1. Additional examples of the input language instruction, scene, and the target output in the ReferIt3D 3D-REC task.

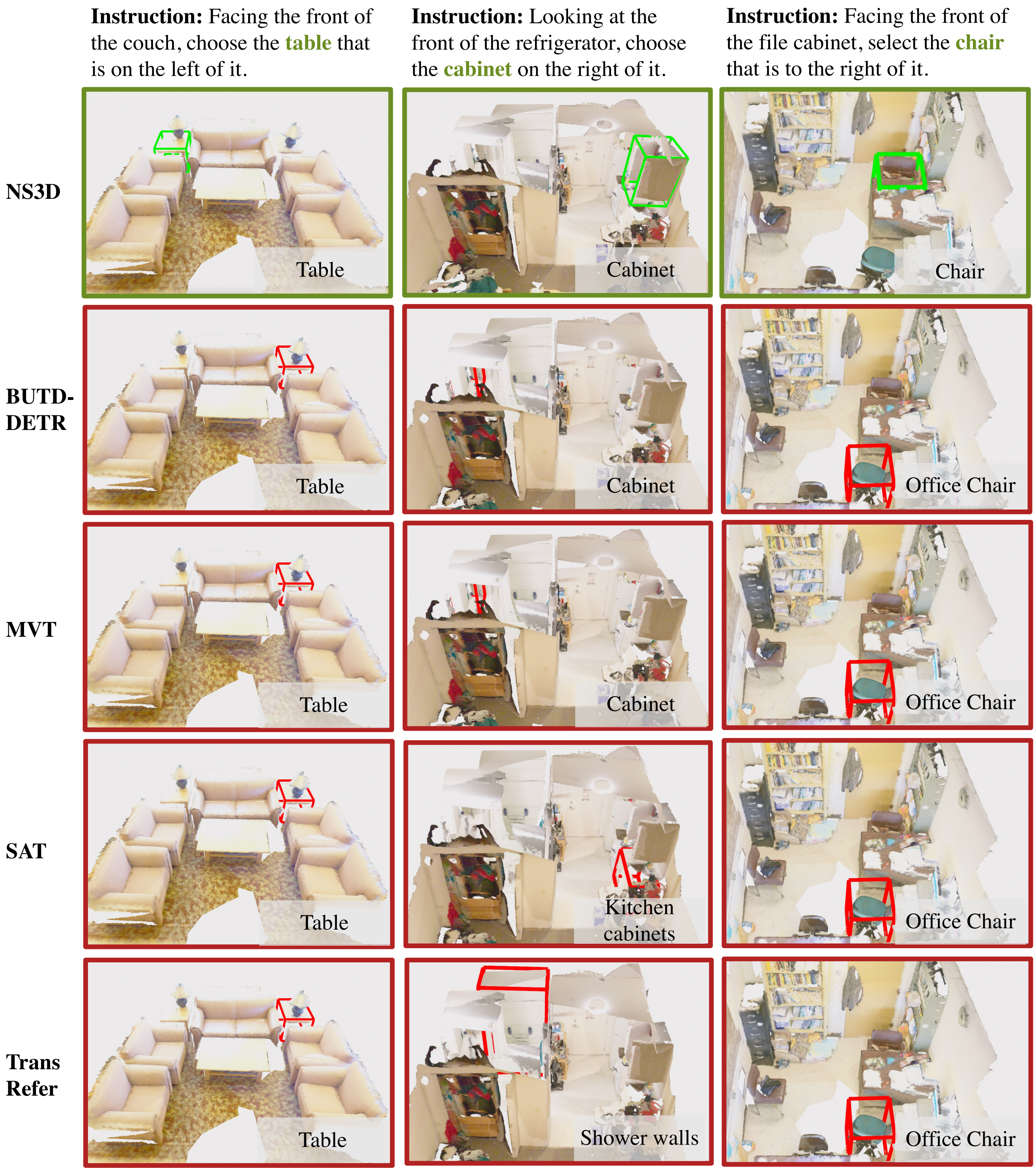


Figure 2. Comparison between NS3D and baselines on the 3D-REC task, with the green outline indicating correct selection and red outline indicating incorrect selection.

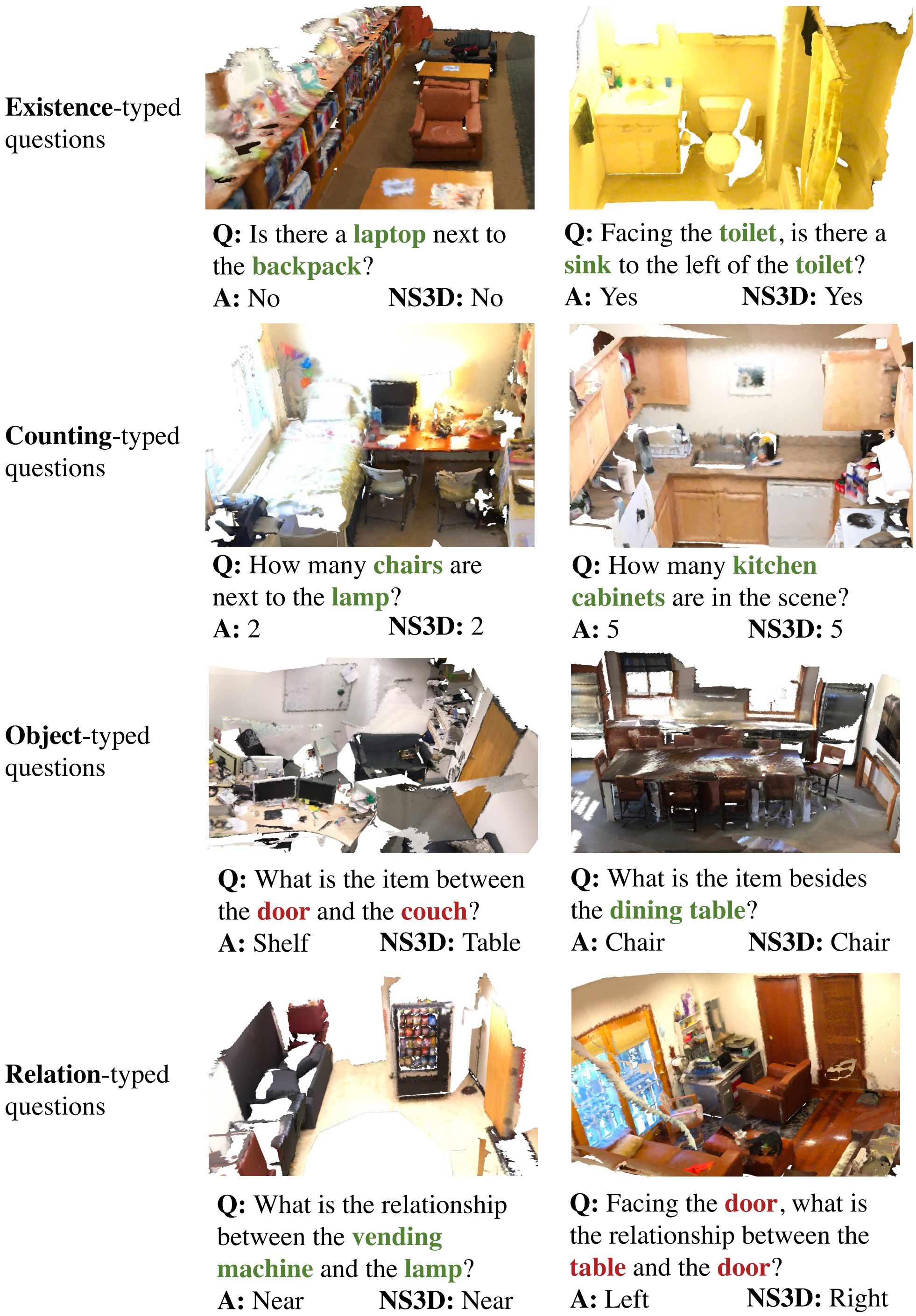


Figure 3. Qualitative examples of NS3D on the 3D-QA task. Examples of success cases are marked in green, while failure cases are marked in red.

References

[1] Panos Achlioptas, Ahmed Abdelreheem, Fei Xia, Mohamed Elhoseiny, and Leonidas Guibas. Referit3d: Neural listeners for fine-grained 3d object identification in real-world scenes. In *European Conference on Computer Vision*, pages 422–440. Springer, 2020. 3

[2] Dailan He, Yusheng Zhao, Junyu Luo, Tianrui Hui, Shaofei Huang, Aixi Zhang, and Si Liu. Transrefer3d: Entity-and-relation aware transformer for fine-grained 3d visual grounding. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 2344–2352, 2021. 4

[3] Shijia Huang, Yilun Chen, Jiaya Jia, and Liwei Wang. Multi-view transformer for 3d visual grounding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15524–15533, 2022. 4

[4] Ayush Jain, Nikolaos Gkanatsios, Ishita Mediratta, and Katerina Fragkiadaki. Bottom up top down detection transformers for language grounding in images and point clouds. In *European Conference on Computer Vision*, pages 417–433. Springer, 2022. 4

[5] Zhengyuan Yang, Songyang Zhang, Liwei Wang, and Jiebo Luo. Sat: 2d semantics assisted training for 3d visual grounding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1856–1866, 2021. 4