Error-Driven Scene Editing for 3D Grounding in Large Language Models

Yue Zhang[1] Zun Wang[1] Han Lin[1] Jialu Li[1] Jianing Yang[2] Yonatan Bitton[3]
Idan Szpektor[3] Mohit Bansal[1]
[1]UNC Chapel Hill [2]University of Michigan [3]Google Research

Abstract

*Despite recent progress in 3D-LLMs, they remain limited in accurately grounding language to visual and spatial elements in 3D environments. This limitation stems in part from training data that focuses on language reasoning rather than spatial understanding due to scarce 3D resources, leaving inherent grounding biases unresolved. To address this, we propose 3D scene editing as a key mechanism to generate precise visual counterfactuals that mitigate these biases through fine-grained spatial manipulation, without requiring costly scene reconstruction or large-scale 3D data collection. Furthermore, to make these edits targeted and directly address the specific weaknesses of the model, we introduce **DEER-3D**, an error-driven framework following a structured "Decompose, Diagnostic Evaluation, Edit, and Re-train" workflow, rather than broadly or randomly augmenting data as in conventional approaches. Specifically, upon identifying a grounding failure of the 3D-LLM, our framework first diagnoses the exact predicate-level error (e.g., attribute or spatial relation). It then executes minimal, predicate-aligned 3D scene edits, such as recoloring or repositioning, to produce targeted counterfactual supervision for iterative model fine-tuning, significantly enhancing grounding accuracy. We evaluate our editing pipeline across multiple benchmarks for 3D grounding and scene understanding tasks, consistently demonstrating improvements across all evaluated datasets through iterative refinement. DEER-3D underscores the effectiveness of targeted, error-driven scene editing in bridging linguistic reasoning capabilities with spatial grounding in 3D LLMs. Our code [1] is publicly available.*

1. Introduction

Grounding language in 3D environments, identifying the specific objects, attributes, and spatial relations that a description refers to, is fundamental for embodied AI and robot manipulation [12, 20, 33, 61]. Recent efforts have integrated large language models (LLMs) with 3D perception, giving rise to so-called *3D-LLMs* [13, 19, 21, 22, 31, 48, 50, 62]. These models have demonstrated impressive progress in open-vocabulary captioning, question answering, and 3D scene reasoning, largely benefiting from the strong linguistic capabilities of LLMs. However, despite these achievements, the grounding performance of current 3D-LLMs remains limited, failing to localize fine-grained visual details (e.g., confusing two pillows of different colors), misinterpreting spatial relations (e.g., confusing "far from" as "near"), or rely on language priors rather than geometric evidence (e.g., grounding "pillow" on the "bed" even when the instruction describes a pillow on the floor) [23, 59]. This gap highlights that linguistic proficiency alone does not guarantee spatially grounded understanding.

Understanding why current 3D-LLMs struggle with grounding requires examining the biases and errors in the model's training. Due to the scarcity of large-scale 3D resources, existing datasets often contain latent statistical biases that models exploit as shortcuts [59]. Fig. 2 illustrates an example that over half of the spatial relationships between "lamp" and "pillow" are categorized as "near," and most pillows are labeled as "white." Consequently, models learn superficial associations (e.g., "white pillows are near lamps") rather than genuine geometric relationships, leading to failures such as incorrectly grounding "a green pillow far from the lamp." However, current approaches for improving 3D-LLMs primarily focus on text-based augmentations [22, 24, 31, 62], which inherently cannot resolve such visual biases. While such strategies improve general linguistic understanding, they leave the underlying 3D scenes unchanged and fail to introduce the necessary visual variations to challenge these spurious co-occurrence patterns, and can even exacerbate them by reinforcing the statistical priors. Moreover, conventional augmentations are typically applied "blindly," without explicitly targeting the model's known grounding errors, resulting in inefficient and ineffective training where the model repeatedly encounters data irrelevant to its weaknesses.

To address these limitations, we propose **DEER-3D** (**D**ecompose, Diagnostic **E**valuation, **E**dit, and **R**etrain), a

[1] https://github.com/zhangyuejoslin/Deer-3D

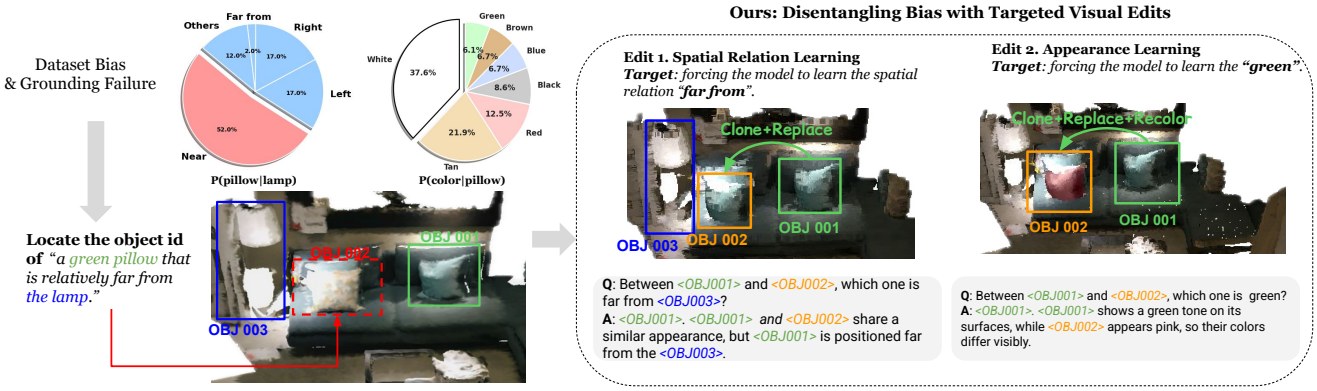


Figure 1. 3D-LLMs frequently overfit to dataset co-occurrence biases (e.g., white pillows), causing grounding failures. We mitigate these biases through targeted counterfactual edits in 3D scenes and construct aligned QA pairs to strengthen the model's grounding ability.

closed-loop framework that introduces visual and spatial variability into 3D-LLMs training through targeted scene editing based on the model's failures. DEER-3D aims to bridge the gap between linguistic understanding and geometric grounding by exposing models to explicit counterfactual changes in 3D object attributes and spatial relations. Instead of heavily relying on textual data, the framework enhances supervision directly in the visual domain by modifying scenes in a minimal yet meaningful way, such as recoloring or repositioning objects. Importantly, these edits are guided by the model's own failure patterns, ensuring that the introduced variability is both relevant and aligned with real grounding errors. Through iterative retraining on these visually augmented examples, DEER-3D strengthens spatial reasoning and encourages models to ground language in visual evidence rather than linguistic shortcuts.

Specifically, for a specific grounding failure, such as incorrectly identifying a bounding box given a description, the DEER-3D framework systematically proceeds through three stages (see Fig. 2). In the **Decompose stage** (Fig. 2(a)), DEER-3D breaks down the natural-language query into atomic predicates that separately describe texture attributes and spatial relations. In the **Diagnose stage** (Fig. 2(b)), it leverages the open-vocabulary reasoning capability of a 3D-LLM to precisely pinpoint predicate-level errors, identifying which semantic factor (e.g., color, orientation, or distance) caused the failure. In the subsequent **Edit stage** (Fig. 2(c)), DEER-3D performs a duplicate-and-replace operation: it duplicates the ground-truth object geometry and substitutes a nearby distractor with this duplicate, ensuring that only the target attribute is edited while all other factors remain fixed. A minimal visual modification, such as recoloring or rotating the duplicate, creates a counterfactual pair that explicitly isolates the erroneous appearance or spatial relation, providing fine-grained supervision signals. We further pair these edited scenes with corresponding question-answer pairs that capture different levels of reasoning difficulty. Finally, in the **Retrain stage** (Fig. 2(d)), these targeted counterfactual examples are integrated back into the training set. Through iterative refinement, the model learns by iteratively correcting its own errors, progressively improving its robustness and predicate-specific grounding accuracy.

We extensively evaluate DEER-3D on established 3D visual grounding benchmarks and observe substantial improvements across all metrics (e.g., achieving 4-5% gains in grounding accuracy), clearly demonstrating its effectiveness in resolving fine-grained grounding errors. Crucially, we validate the benefit of our iterative refinement framework by conducting multi-round experiments, illustrating how progressively targeting model failures further boosts performance. Comprehensive ablation studies confirm the effectiveness of our targeted visual editing strategies, the importance of error-driven counterfactual augmentation, and the impact of different question design choices. Additionally, we evaluate DEER-3D's broader utility beyond grounding, demonstrating consistent improvements on general 3D tasks, as well as on human-aligned grounding evaluations. Taken together, these experiments underscore the effectiveness and robustness of our approach in enhancing the grounding capabilities of 3D-LLMs.

2. Related Work

3D-LLMs for Grounding. Recent advances in 3D-LLMs have greatly expanded the paradigm of 3D grounding by coupling LLMs with 3D visual perceptions, enabling open-vocabulary understanding and spatial reasoning in 3D scene envrionments [10, 13, 19, 22, 24, 30, 48, 56, 57, 62, 64, 66]. Despite their impressive reasoning ability, current 3D-LLMs often suffer from visual bias, relying on linguistic priors rather than genuine 3D evidence when making predictions. A primary reason for this vulnerability is

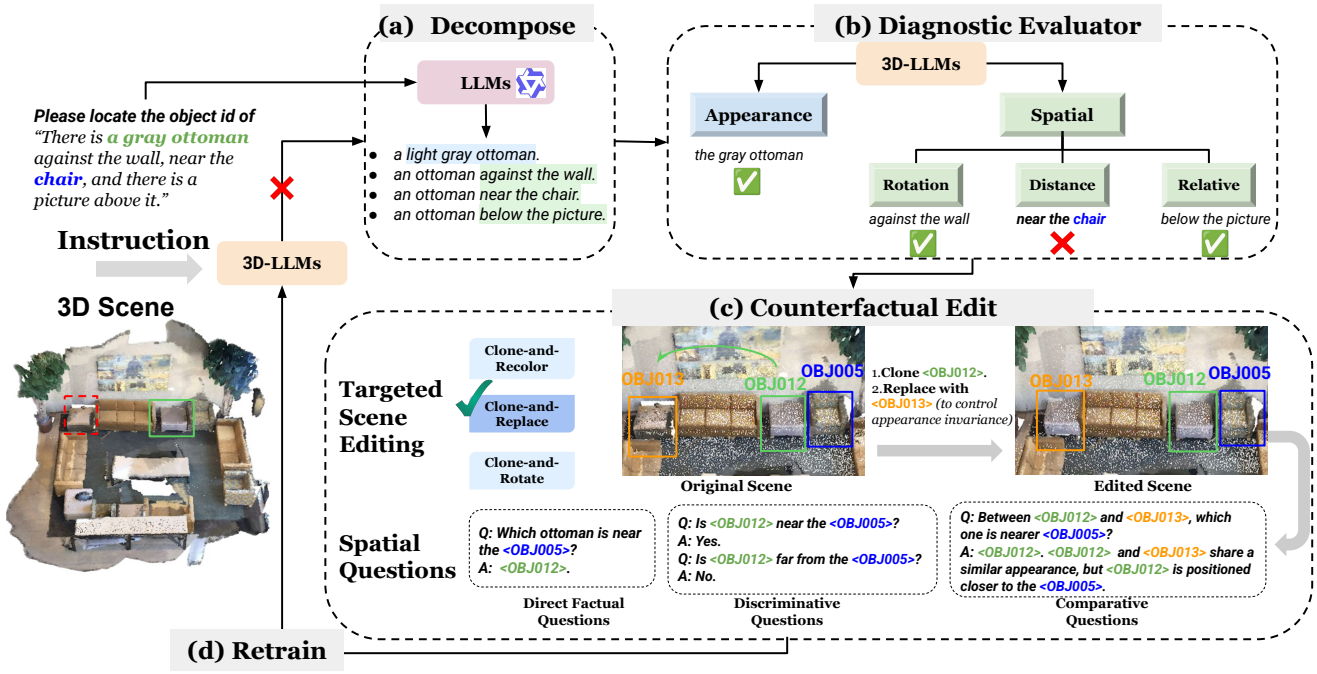


Figure 2. Overview of DEER-3D. When a grounding error is detected, DEER-3D performs targeted visual edits. Given a natural-language instruction, the framework (a) decomposes it into atomic predicates, (b) diagnoses the specific error, and (c) applies predicate-level visual edits. Aligned question–answer pairs are then created to explicitly supervise the failed predicate. Finally, the model (d) iteratively retrains on these counterfactual examples to progressively improve grounding accuracy.

the community's reliance mainly on text-based augmentation to compensate for data scarcity. Methods such as rephrasing, caption generation, or synthetic instruction creation [22, 31, 62] improve linguistic diversity but leave the underlying 3D visual data unchanged. This approach not only fails to correct existing dataset biases but can inadvertently exacerbate them by reinforcing the model's reliance on spurious linguistic correlations. In contrast, we address this limitation from the visual side through an error-driven visual augmentation framework that explicitly edits 3D scenes to generate counterfactual pairs targeting the exact predicates responsible for grounding failures, providing fine-grained supervision to correct visual bias.

Counterfactual Augmentation. Counterfactual data augmentation has been widely adopted across multiple tasks to enhance model robustness, fairness, and causal understanding. It has been applied for debiasing and improving generalization through minimal text rewrites in NLP [17, 25, 34, 68], disentangling causal visual factors via counterfactual image synthesis in vision [35, 38, 43], improving seen-unseen gap in robotics [16, 29, 37, 46], and strengthening compositional and causal reasoning in multimodal tasks such as visual question answering [8, 27, 41]. However, most existing efforts remain limited to 2D or textual domains. These methods are fundamentally insufficient for the unique challenges of 3D grounding, as they are not designed to perform targeted, predicate-level interventions on complex 3D geometric properties such as object orientation or distance. In contrast, we introduce model-diagnosed, visually grounded counterfactual editing in 3D scenes to improve grounding ability.

Error-Driving Learning. Error-driven and self-corrective learning has been applied in NLP [2, 26, 45, 54, 58], multimodal models [47, 55], and robotics [14], where models improve by learning from corrected outputs [2, 42, 51] or retraining on targeted failure cases [1, 14, 47]. While prior work relies on textual feedback or task-level supervision, our framework instead performs instance-level correction through an iterative 3D visual-domain loop that identifies grounding failures and applies minimal scene edits to progressively enhance spatial understanding in 3D-LLMs.

3D Scene Editing. Although recent 3D scene editing methods [18, 28, 40, 49, 52, 67] enable high-fidelity and interactive object manipulation, their design goals differ fundamentally from the requirements of our counterfactual training pipeline. These systems prioritize photorealistic rendering and user-driven editing, rather than producing semantically controlled, single-factor modifications that preserve all other scene attributes. In contrast, DEER-3D requires large-scale, reproducible, and predicate-isolated interven-

tions, while keeping the rest of the scene unchanged.

3. Method

In this section, we introduce our approach, which systematically identifies and corrects grounding errors through targeted counterfactual scene editing. As illustrated in Fig. 2, our framework is structured as follows: we first describe our baseline model (Sec. 3.1). Next, we present the core components of our pipeline, beginning with the decomposition of complex instructions into simpler, atomic sub-descriptions, each capturing a distinct semantic factor (Sec. 3.2). We then introduce a diagnostic evaluator that pinpoints the model's specific reasoning failures related to these semantic factors (Sec. 3.3).Then, we detail our scene-editing strategy (Sec. 3.4), which generates controlled counterfactual 3D scenes along with aligned question-answer pairs. Finally, we describe how the base model is iteratively fine-tuned using the collected counterfactual data (Sec. 3.5), progressively enhancing its grounding accuracy.

3.1. Base 3D-LLM Model

We build our pipeline upon a 3D-LLM, denoted as M_{base}, which serves as our initial grounding model. Specifically, we utilize ChatScene [21] as the backbone. In the grounding task, the base model receives inputs consisting of 3D object points S with color and instance segmentation obtained by 3D segmentor [39], along with a natural language instruction T describing a target object. The model processes these inputs to produce a prediction O_{pred} (a 3D bounding box) that localizes the object specified by the instruction:

$$O_{pred} = M_{base}(S, T),$$

This initial prediction O_{pred} is then evaluated against the ground truth. We detail our error-driven analysis framework in the following sections.

3.2. Instruction Decomposition

Our DEER-3D framework is activated only when the base model's prediction O_{pred} is incorrect. To precisely diagnose errors, we first decompose complex, free-form instructions T into their atomic components (sub-instruction). As shown in Fig. 2(a), we employ a large language model (*e.g.*, Qwen3 [53]) to systematically parse T into a set of atomic predicates, denoted as $\mathbf{P} = \{p_1, p_2, \cdots, p_n\}$. This decomposition is crucial, as it isolates potential error sources to individual semantic elements of the original instruction. For example, given the compositional instruction T = *"the brown couch against the wall"*, the parser typically outputs atomic predicates capturing distinct semantic components such as object appearance and spatial relations (e.g., p_1 = *"the brown couch"*, p_2 = *"the couch is against the wall"*. The set of atomic predicates $\mathbf{P}$ is then forwarded to our diagnostic evaluator introduced as follows. Detailed prompts used for this decomposition are provided in Appendix A.1.

3.3. Diagnostic Evaluator

Given the set of atomic predicates $\mathbf{P} = \{p_1, ..., p_n\}$, the goal of the diagnostic evaluator is to pinpoint which specific predicate p_i the model failed to ground (shown in Fig. 2(b)). From empirical observations, we identified two primary types of visual grounding errors. A detailed analysis of these error categories and their distributions is provided in Appendix A.2, where we show that these two types collectively account for over +75% of all errors in our experiments. We categorize primary error types as follows:

- **Appearance Grounding**: Descriptions related to visual appearance, particularly textures and colors (*e.g., "brown/wooden couch"*).
- **Spatial Grounding**: Descriptions specifying the spatial context, including **orientation** (*e.g.*, "*against the wall*"), **distance** (*e.g.*, "*near the ottoman*"), or **relative positioning** (*e.g.*, "*below the picture*").

To perform this diagnosis, each atomic sub-instruction (predicate) $p_i \in \mathbf{P}$ is individually queried against the model M_{base}. The model's task is to generate a set of candidate object IDs $\{O_{cand}\}$ that satisfy the specific predicate p_i. We then check if the ground-truth object O_{gt} is included in this set: $O_{gt} \in \{O_{cand}\}$. If $O_{gt} \notin \{O_{cand}\}$, we identify p_i as a grounding failure. Finally, the identified error type determines which counterfactual editing strategy to apply in the subsequent augmentation stage.

3.4. Error-Driven Counterfactural Augmentation

Our method generates 3D counterfactual scenes through precise, error-specific modifications. Specifically, as shown in Fig. 2(c), when a visual grounding error is detected, our goal is to isolate the semantic factor responsible for the error while keeping all other visual and geometric properties constant. To achieve this, we construct **controlled counterfactual scenes** that differ from the original scene along only one visual predicate. This enables controlled supervision so that the model must correctly attend to the changed attribute to succeed. Our editing framework follows a unified **Clone–Replace–Modify** procedure:

- **Clone:** Let o_{gt} be the ground-truth object. We first create o_{clone}, a perfect duplicate of o_{gt} with identical geometry, material, and texture properties.
- **Replace:** Let o_d be a distractor object, strategically selected based on the diagnosed error type to create a challenging counterfactual example. We remove o_d from the scene and place o_{clone} at the exact 3D position, T_d, previously occupied by o_d. Formally, this operation is simply:

$$o_{\texttt{clone}}^{\texttt{relocated}} = \texttt{Translate}(o_{\texttt{clone}}, T_d),$$

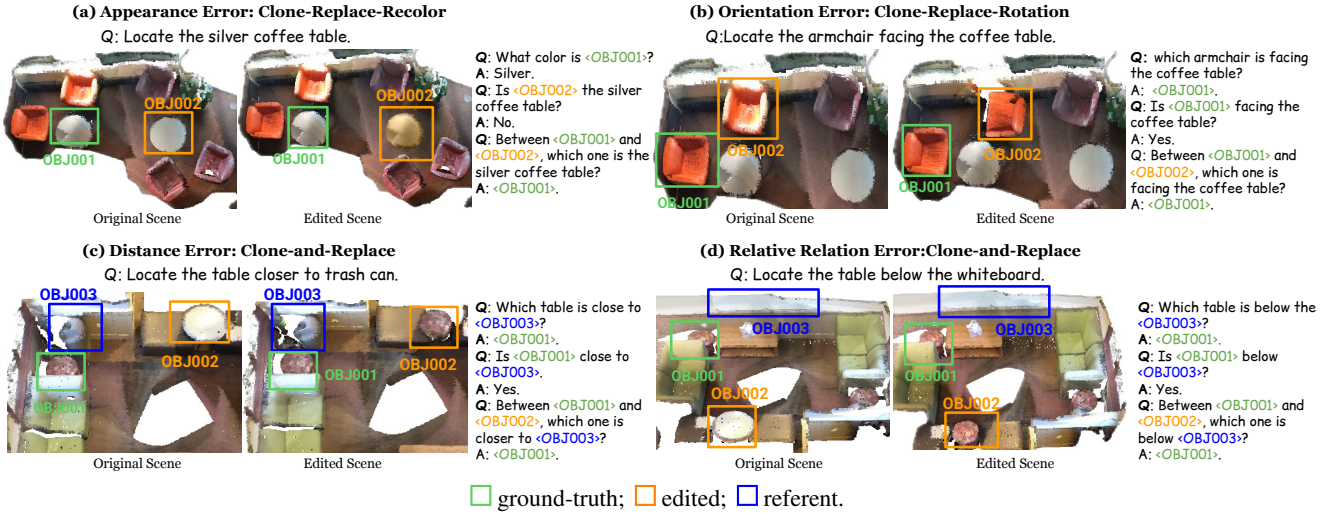


Figure 3. Examples of targeted visual edits for different grounding errors. Each edit generates aligned question–answer pairs, precisely supervising the model's visual grounding ability.

- **Modify:** We apply a predicate-specific modification to the newly placed o_{clone}. This modification directly corresponds to the diagnosed error type that caused the failure.

This unified structure allows the same editing principle to address different visual error categories. In the following, we describe the specific editing operations designed for each visual error type.

3.4.1. Error-Specific Edits

While our unified Clone–Replace–Modify framework provides a general mechanism for controlled counterfactual editing, the **Replace** and the **Modify** step varies according to the semantic predicate causing the grounding error. We provided an example of targeted visual editing for different grounding errors in Fig. 3.

Clone–Replace–Recolor (CR-Rec) for Appearance Errors. For appearance-based grounding errors (e.g., incorrect color identification), we first select a nearby distractor object o_d that shares the same semantic category as the ground-truth object o_{gt} (*e.g.*, coffee table <OBJ002> in Fig. 3(a)). After cloning and replacing as described above, we recolor the cloned object o_{clone} to a perceptually contrasting hue while preserving its original shading and illumination properties. Formally, the edited object $\tilde{o}$ is obtained by sequentially translating and recoloring:

$$\tilde{o} = \texttt{Recolor}\big(\texttt{Translate}(o_{\texttt{clone}}, T_d),\, c^*\big),$$

where T_d is the original 3D position of the distractor, and c^* is the selected contrasting hue that maximizes perceptual difference from the ground-truth color in CIELAB space:

$$c^* = \arg\max_{c \in \mathcal{C}} \|\texttt{Lab}(o_{gt}) - \texttt{Lab}(c)\|_2,$$

where $\texttt{Lab}(o_{gt})$ denotes the original object's mean color in CIELAB space, and c represents a candidate color from a predefined set of distinct candidate hues (more details are provided in Appendix A.3). This controlled recoloring provides two visually identical objects differing in color, enabling precise supervision for appearance grounding.

Clone–Replace–Rotate (CR-Rot) for Orientation Errors. For grounding errors involving orientation predicates (e.g., *"facing the table"*, *"against the wall"*), we first select a nearby distractor object o_d (e.g. the armchair <OBJ002> in Fig. 3(b)) from the same semantic category as the ground-truth object o_{gt}. After cloning and replacing as previously described, we apply a controlled rotation to the cloned object o_{clone}. Formally, the final edited object $\tilde{o}$ is obtained by sequentially translating and rotating:

$$\tilde{o} = \texttt{Rotate}\big(\texttt{Translate}(o_{\texttt{clone}}, T_d),\, R_{\mathbf{y}}(\theta)\big),$$

where $\mathbf{y}$ is the vertical rotation axis and $R_{\mathbf{y}}(\theta)$ denotes a rotation matrix around the Y-axis by angle θ. In our experiments, the rotation angle θ is selected from a small predefined set $\{\pm 45°, \pm 90°\}$ to produce meaningful yet visually consistent orientation changes. This unified formulation explicitly integrates spatial relocation and orientation modification, yielding two visually identical scenes differing only in object orientation, thereby providing precise supervision for directional grounding.

Clone–and–Replace (CR) for Distance Errors. Distance grounding errors involve a **referent object** (e.g., the transh can <OBJ003> in Fig. 3(c)), which defines the spatial relationship of interest. To identify this referent, we first filter all surrounding objects sharing the same semantic label as the mentioned category (e.g., *"tran can"* instances). The specific referent object o_{ref} is then selected according to the spatial predicate in the instruction: For *"near"* relations, we select the nearest instance to the target object as o_{ref}; For *"far"* relations, we select the farthest instance within the

same category as o_{ref}. After determining the referent o_{ref}, we select a distractor o_d of the same category as the ground-truth object o_{gt} to construct the counterfactual pair. The distractor selection follows the relational constraint: For a *"near"* predicate, o_d must be farther from o_{ref} than the ground-truth object, and vice versa for a *"far"* predicate. Finally, we replace the selected distractor with the cloned ground-truth object.

Clone–and–Replace (CR) for Relative Relation Errors. We focus on vertical relations (e.g., *"above"*, *"below"*), since horizontal relative-position annotations in ScanNet (e.g., *"left"*, *"right"*) are frequently noisy and lack consistent labeling [23]. Similar to the CR strategy for distance errors, we first identify the referent object that defines the relation in the instruction (e.g., the whiteboard <OBJ003> in Fig. 3(d)). After identifying o_{ref}, we select a distractor object o_d of the same semantic category as the ground-truth object o_{gt} (e.g., table <OBJ002> in Fig. 3(d)) that violates the intended vertical relation. For instance, when the instruction specifies "below", we select a distractor that is not located below the referent, ensuring a meaningful counterfactual spatial configuration.

3.4.2. QA Generation

After constructing counterfactual scenes containing both the ground-truth object o_{gt} and its edited object $\tilde{o}$, we generate new aligned question–answer (QA) pairs to explicitly target the model's original grounding failure. The objective is to encourage the model to distinguish between o_{gt} and $\tilde{o}$ based solely on the failed predicate. Each counterfactual scene is paired with around 5-6 QA examples with increasing reasoning complexity, categorized as follows:

- **Direct Factual (Perception).** Simple open-ended queries assess the model's perception of each object's attribute. *Example:* Q: "What color is o_{gt}?" → A: "Light green." Q: "What color is $\tilde{o}$?" → A: "Pink."
- **Discriminative (Verification).** Yes/no questions verify whether the model correctly understands the failed predicate for each object. *Example:* Q: "Is o_{gt} light green?" → A: "Yes." Q: "Is $\tilde{o}$ light green?" → A: "No."
- **Comparative (Reasoning).** Comparative queries require the model to directly contrast the minimal pair, isolating the failed attribute through comparison. Each answer identifies the correct object and explicitly states the rationale behind the choice by highlighting differences. *Example:* Q: "Between o_{gt} and $\tilde{o}$, which one is light green?" → A: "o_{gt}. o_{gt} shows a light tone on its surfaces, while o_{gt} appears gold, so their colors differ visibly."

This mixture of QA types provides comprehensive supervision for each counterfactual edit, targeting multiple levels of reasoning difficulty.

Table 1. Performance comparison on 3D grounding benchmarks. Our method consistently outperforms Chat-Scene (w/o 2D) and Chat-Scene on all metrics for ScanRefer and Multi3DRefer, with iterative re-training further improving results.

Method	ScanRefer		Multi3DRefer	
	Acc@0.25	Acc@0.5	F1@0.25	F1@0.5
Expert Models				
ScanRefer [6]	37.3	24.3	-	-
3DJCG [5]	49.6	37.3	-	-
M3DRef-CLIP [60]	51.9	44.7	42.8	38.4
3D-VisTA [65]	50.6	45.5	-	-
LLM-based Models				
3D-LLM [19]	30.3	-	-	-
Chat-3D [48]	35.9	30.4	-	-
Chat-3D v2 [21]	42.5	38.4	45.1	41.6
Grounded 3D-LLM [10]	47.9	44.1	45.2	40.6
PQ3D [66]	57.0	51.2	-	50.1
3D-LLaVA [13]	51.2	40.6	-	-
Inst3D-LLM [56]	57.8	51.6	58.3	53.5
Chat-Scene (w/o 2D) [21]	41.2	37.4	43.8	40.2
Ours (Round 1)	45.5 (+4.3)	41.8 (+4.4)	48.4 (+4.6)	45.1 (+4.9)
Ours (Round 2)	47.1 (+5.9)	43.1 (+5.7)	50.3 (+6.5)	47.1 (+6.9)
Chat-Scene [21]	55.5	50.2	57.1	52.4
Ours (Round 1)	57.8 (+2.3)	52.3 (+2.1)	60.0 (+2.9)	55.8 (+3.4)
Ours (Round 2)	**58.6** (+3.1)	**53.3** (+3.1)	**61.4** (+4.3)	**56.8** (+4.4)

3.5. Iterative Re-training

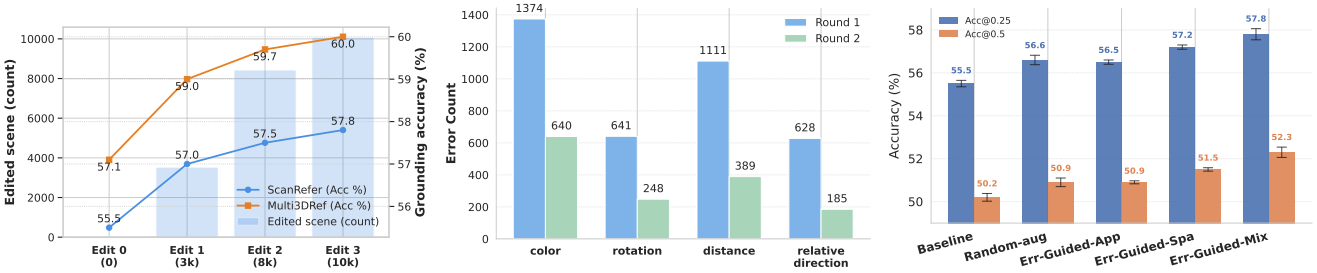
The re-training phase shown in Fig. 2(d) completes the DEER-3D loop. Specifically, we combine the original training data with newly generated counterfactual scene and QA pairs. The base model M_{base} is fine-tuned on this combined dataset to obtain a refined model with improved grounding performance on the previously identified errors. This refined model subsequently serves as the baseline for the next iteration, where it is employed to mine remaining failures and generate additional counterfactuals. By iteratively repeating this process, DEER-3D progressively enhances the model's grounding accuracy.

4. Experiments

In this section, we evaluate our method on standard 3D grounding benchmarks. We adopt two backbone variants of the state-of-the-art **Chat-Scene** baseline [21]: (1) **Chat-Scene (3D-only)**, which utilizes textual instructions and 3D point clouds without 2D visual inputs, and (2) **Chat-Scene**, the original version that additionally incorporates complementary 2D multi-view images. This setup enables us to assess our approach under different modality conditions.

4.1. Datasets and Evaluation Metrics

Dataset. We evaluate our method on standard 3D referring expression grounding benchmark: ScanRefer [6] and Multi3DRefer [60]. ScanRefer provides single-object referring expressions, testing a model's ability to localize fine-grained visual and spatial attributes. Multi3DRefer builds upon this by introducing multi-object references and re-



(a) Edit scaling effects. (b) Error counts across iterations. (c) Different types of visual edits.

Figure 4. Ablation studies validating our method. (a) Grounding performance scales with the quantity of our counterfactual data. (b) Our iterative loop (Round 2 vs. Round 1) successfully reduces all targeted error types. (c) Our full Err-Guided-Mix strategy is superior to both Random-aug and individual components, validating our error-driven and complementary design.

Table 2. Ablation on different QA types: factual (Fact.), discriminative (Discri.), comparative without explanations (Comp.*), and comparative with explanations (Comp.).

#	QA Types				Multi3DRefer	
	Fact.	Discri.	Comp.*	Comp.	Acc@0.25	Acc@0.5
1					57.1	52.4
2	✓				57.8	53.0
3	✓	✓			58.3	53.6
4	✓		✓		58.9	54.0
5	✓	✓	✓		59.1	54.2
Ours	✓	✓		✓	**60.0**	**55.8**

lational reasoning, where multiple objects must be jointly grounded within the same 3D scene.

Evaluation Metrics. We adopt standard evaluation metrics specific to each benchmark. For grounding tasks on ScanRefer [6] and Multi3DRefer [60], we report grounding accuracy (Acc) at Intersection over Union (IoU) thresholds of 0.25 (Acc@0.25) and 0.5 (Acc@0.5), where predictions are considered correct if their IoU with ground-truth annotations exceeds the corresponding threshold. For multi-object grounding on Multi3DRefer [60], we report F1 scores at these IoU thresholds (F1@0.25, F1@0.5).

4.2. Implementation Details

In Round 1, we fine-tune the baseline model using the original training data plus 10k counterfactual scenes and 60k QA pairs. In Round 2, we use the Round 1 model to identify new failure cases, generate an additional 3k counterfactual scenes and 18k QA pairs, and further fine-tune the model to obtain the Round 2 model. For all fine-tuning stages, we train for 5 epochs using the AdamW optimizer. The plearning rate is $3e^{-5}$, with a weight decay of 0.01 and a global batch size of 32. All experiments are conducted on 4 NVIDIA H100 (80 GB) GPUs. More implementation details are provided in Appendix B.1.

4.3. Experimental Results

Grounding Performance. As shown in Table 1, our approach consistently outperforms prior models across both 3D grounding benchmarks. While traditional expert models rely on task-specific architectures, our goal is to enhance *LLM-based* models that jointly reason over language and 3D scenes. Results demonstrate significant performance gains, particularly notable in the purely 3D setting, where our counterfactual augmentation delivers improvements of up to around $+5\%$ in the first round of augmentation alone. This highlights our method's effectiveness in directly enhancing spatial and attribute grounding from 3D signals. Moreover, when combined with the richer multimodal inputs provided by Chat-Scene, our approach achieves further improvements, establishing a new state-of-the-art performance across both datasets. These consistent gains underscore the broad applicability and robustness of our counterfactual augmentation strategy in significantly enhancing grounded understanding in 3D-LLMs.

Error-Guided Iterative Refinement. We further validate the effectiveness of our error-guided iterative editing loop, as shown in Table 1. Starting from the initial improvement achieved by the first round of counterfactual augmentation (Round 1), we apply a subsequent iteration (Round 2), producing additional performance gains across both benchmarks and baselines. Crucially, these iterative gains are consistent for both purely 3D and multimodal baselines, highlighting that our approach is universally beneficial regardless of input modalities. This clearly demonstrates that our iterative loop is more than just a one-time data enhancement; instead, it serves as a sustainable self-correction mechanism, progressively refining the model by systematically addressing its weaknesses. More detailed analyses of each iterative step are provided in our ablation study.

4.4. Ablation Study

In this section, we present ablation studies conducted with Chat-Scene with 2D images as our baseline model.

Effect of Augmentation Scale. We analyze how grounding performance scales with the amount of counterfactual data generated by varying the number of edited distractors around each ground-truth object. Specifically, in Fig.4a,

Table 3. Experimental results on scene understanding tasks. Our method outperforms baselines and shows strong overall capability across general scene understanding tasks.

Method	Scan2Cap		ScanQA		SQA3D	
	C@0.5	B-4@0.5	C	B-4	EM	EM-R
3D-LLM [19]	-	-	69.4	12.0	-	-
Chat-3D v2 [48]	63.9	31.8	87.6	14.0	54.7	-
LL3DA [9]	65.2	36.8	76.8	13.5	-	53.7
3D-LLaVA [13]	78.7	36.9	92.6	-	54.5	-
LEO [22]	68.4	**36.9**	80.0	11.5	-	53.7
SceneLLM [15]	-	-	80.0	12.0	-	54.2
Chat-Scene (w/o 2D) [21]	64.9	–	80.3	–	53.4	-
Ours	72.4	35.0	88.7	14.3	53.6	56.3
Chat-Scene [21]	77.1	36.3	87.7	**14.3**	54.6	57.5
Ours	**81.4**	35.8	**89.5**	13.1	**56.3**	**58.9**

each increment in the "Edit" setting (Edit-0 → Edit-3) corresponds to progressively increasing the sampling of distractors per scene (e.g., from 1 to 3, then 5), resulting in a cumulative increase in total edited scenes. Performance on both ScanRefer and Multi3DRefer benchmarks improves consistently with increased sampling, validating the benefit of richer and more diverse counterfactual edits. Notably, the largest gains occur during initial scaling (Edit-0→Edit-1), suggesting that sampling a small set of distractors already addresses many critical grounding issues effectively. Subsequent increments continue to provide steady but diminishing returns, highlighting a balance between dataset richness and augmentation efficiency.

Iterative Refinement Analysis. In Table. 1, we have presented quantitative results demonstrating the effectiveness of our error-guided iterative refinement strategy in Table.1. To further analyze its impact, Fig.4b explicitly compares grounding error counts between Round 1 and Round 2 across predicate categories. We observe substantial reductions in errors for all predicate types, clearly indicating that our framework effectively identifies and addresses the model's primary grounding failures, thereby progressively refining its visual and spatial grounding capabilities. Additionally, we conduct a third iteration (Round 3 results included in the Appendix B.3, using the Round 2 model to identify remaining failure cases. However, further training resulted in almost the same results as the Round 2, indicating saturation in grounding performance beyond Round 2. Thus, the Round 2 model represents an optimal trade-off between grounding accuracy and computational efficiency.

Ablation on Counterfactual Edit Types. To analyze the effectiveness of each edit type within our counterfactual augmentation framework, we conduct an ablation study as shown in Fig.4c. Starting from the baseline, a simple Random-aug strategy, which applies edits randomly without considering specific grounding failures (keeping the total number of edits equal to our full strategy), already yields a modest gain. This indicates that generic augmentation alone can slightly enhance model performance. In contrast, our targeted strategies are far more efficient and effective. Notably, our Err-Guided-App (appearance-only) achieves a comparable performance despite using significantly less data than the full Random-aug set. Furthermore, our Err-Guided-Spa (spatial-only augmentation) not only uses less data than Random-aug but also outperforms it. Finally, combining these two targeted edit types into Err-Guided-Mix leads to the highest performance, clearly demonstrating the effectiveness of our counterfactual edits.

Table 4. Experimental results on Beacon3D [23]. "App.", "Geo.", and "Spa." denote appearance, geometry, and spatial relations, respectively. "Overall (Case)" reports average accuracy per case, while "Overall (Obj)" reports average accuracy per object.

Model	Class	App.	Geo.	Spa.	Overall (Case)	Overall (Obj)
Chat-Scene [24]	59.8	59.2	50.2	53.8	59.8	41.0
Ours	**61.7**	**64.4**	**53.4**	**55.5**	**61.7**	**45.7**

Ablation on Different Question Types. We further examine the contribution of each QA supervision type (Table. 2). Starting from the baseline (Row 1), progressively adding factual (Row 2), discriminative (Row 3), and comparative questions (Row 4) consistently improves model performance by guiding it to address specific and challenging grounding queries. To isolate the effect of explanations, Row 5 includes comparative questions without explicit rationales, whereas ours incorporate full comparative questions with explanations. The performance gap between these settings shows that explanations provide an additional boost, indicating that explicit reasoning about the differences further strengthens grounding.

4.5. Other Analysis

Performance on Captioning and QA. We further assess the generalization of our grounding-focused augmentation on broader scene understanding tasks, including captioning and question answering (Scan2Cap [11], ScanQA [3], and SQA3D [32]). As shown in Table 3, our method achieves consistent improvements over the baseline across all benchmarks, indicating that the benefits of our approach extend beyond grounding-specific settings. Detailed metric definitions and additional analysis are provided in Appendix B.2.

Evaluation of Error-driven Framework. To evaluate the effectiveness of our error-guided diagnostic framework without visual edits, we conduct an experiment using a text-only strategy ("DEER-3D + Text-Aug"). Specifically, we leverage our diagnostic pipeline to pinpoint ambiguous errors, but instead of visually editing the 3D scenes, we generate clearer textual instructions that explicitly disambiguate each failure case. For instance, if the model confuses two visually similar radiators, our framework identifies this ambiguity and generates a more precise instruction (e.g., "*locate the radiator to the north*"), while leaving the scene unchanged. Training solely on this text-augmented data

already improves performance over the baseline (55.5% → 56.6% Acc@0.25 on ScanRefer). This demonstrates that inherent value of our diagnostic approach contributes to model robustness, even without visual augmentation. Please refer to the Appendix B.4 for more details.

Evaluation on Human-annotated 3D Grounding. To further validate the real-world reliability of our method, we evaluate on the human-annotated BEACON-3D benchmark [23], which provides manually verified object and relation annotations for 3D scenes. This dataset reflects human judgments of spatial and visual grounding, offering a more rigorous test of perceptual understanding. As shown in Table. 4, our model outperforms Chat-Scene across all categories, indicating that our error-driven augmentation not only enhances quantitative performance but also yields more human-aligned grounding behavior.

5. Conclusion

We introduce DEER-3D, an error-driven framework that enhances 3D grounding via targeted visual counterfactual editing. Rather than random textual augmentation, DEER-3D diagnoses grounding errors, decomposes them into semantic factors, and generates controlled 3D edits for precise predicate-level supervision. Through iterative refinement, our approach consistently improves grounding performance. Comprehensive experiments and analysis further demonstrate that error-guided scene editing effectively strengthens grounding capabilities in 3D-LLMs.

6. Acknowledgements

We thank Jaemin Cho, Elias Stengel-Eskin, and Zaid Khan for their helpful feedback. This work was supported by NSF-AI Engage Institute DRL-2112635, ARO Award W911NF2110220, ONR Grant N00014-23-1-2356, DARPA ECOLE Program No. HR00112390060, and a Capital One Research Award. The views contained in this article are those of the authors and not of the funding agency.

References

[1] Afra Feyza Akyürek, Ekin Akyürek, Ashwin Kalyan, Peter Clark, Derry Tanti Wijaya, and Niket Tandon. Rl4f: Generating natural language feedback with reinforcement learning for repairing model outputs. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7716–7733, 2023. 3

[2] Shengnan An, Zexiong Ma, Zeqi Lin, Nanning Zheng, Jian-Guang Lou, and Weizhu Chen. Learning from mistakes makes llm better reasoner. *arXiv preprint arXiv:2310.20689*, 2023. 3

[3] Daichi Azuma, Taiki Miyanishi, Shuhei Kurita, and Motoaki Kawanabe. Scanqa: 3d question answering for spatial scene understanding. In *proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 19129–19139, 2022. 8, 2

[4] Daigang Cai, Lichen Zhao, Jing Zhang, Lu Sheng, and Dong Xu. 3djcg: A unified framework for joint dense captioning and visual grounding on 3d point clouds. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16443–16452, 2022. 2

[5] Daigang Cai, Lichen Zhao, Jing Zhang, Lu Sheng, and Dong Xu. 3djcg: A unified framework for joint dense captioning and visual grounding on 3d point clouds. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16464–16473, 2022. 6

[6] Dave Zhenyu Chen, Angel X Chang, and Matthias Nießner. Scanrefer: 3d object localization in rgb-d scans using natural language. In *European conference on computer vision*, pages 202–221. Springer, 2020. 6, 7

[7] Dave Zhenyu Chen, Qirui Wu, Matthias Nießner, and Angel X. Chang. D3net: A speaker-listener architecture for semi-supervised dense captioning and visual grounding in rgb-d scans. *ArXiv*, abs/2112.01551, 2021. 2

[8] Long Chen, Xin Yan, Jun Xiao, Hanwang Zhang, Shiliang Pu, and Yueting Zhuang. Counterfactual samples synthesizing for robust visual question answering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10800–10809, 2020. 3

[9] Sijin Chen, Xin Chen, Chi Zhang, Mingsheng Li, Gang Yu, Hao Fei, Hongyuan Zhu, Jiayuan Fan, and Tao Chen. Ll3da: Visual interactive instruction tuning for omni-3d understanding reasoning and planning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 26428–26438, 2024. 8

[10] Yilun Chen, Shuai Yang, Haifeng Huang, Tai Wang, Runsen Xu, Ruiyuan Lyu, Dahua Lin, and Jiangmiao Pang. Grounded 3d-llm with referent tokens. *arXiv preprint arXiv:2405.10370*, 2024. 2, 6

[11] Zhenyu Chen, Ali Gholami, Matthias Nießner, and Angel X Chang. Scan2cap: Context-aware dense captioning in rgb-d scans. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3193–3203, 2021. 8, 2

[12] An-Chieh Cheng, Hongxu Yin, Yang Fu, Qiushan Guo, Ruihan Yang, Jan Kautz, Xiaolong Wang, and Sifei Liu. Spatialrgpt: Grounded spatial reasoning in vision-language models. *Advances in Neural Information Processing Systems*, 37:135062–135093, 2024. 1

[13] Jiajun Deng, Tianyu He, Li Jiang, Tianyu Wang, Feras Dayoub, and Ian Reid. 3d-llava: Towards generalist 3d lmms with omni superpoint transformer. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 3772–3782, 2025. 1, 2, 6, 8

[14] Jiafei Duan, Wilbert Pumacay, Nishanth Kumar, Yi Ru Wang, Shulin Tian, Wentao Yuan, Ranjay Krishna, Dieter Fox, Ajay Mandlekar, and Yijie Guo. Aha: A vision-language-model for detecting and reasoning over failures in robotic manipulation. *arXiv preprint arXiv:2410.00371*, 2024. 3

[15] Rao Fu, Jingyu Liu, Xilun Chen, Yixin Nie, and Wenhan Xiong. Scene-llm: Extending language model for 3d visual understanding and reasoning. *arXiv preprint arXiv:2403.11401*, 2024. 8

[16] Tsu-Jui Fu, Xin Eric Wang, Matthew F Peterson, Scott T Grafton, Miguel P Eckstein, and William Yang Wang. Counterfactual vision-and-language navigation via adversarial path sampler. In *European Conference on Computer Vision*, pages 71–86. Springer, 2020. 3

[17] Matt Gardner, Yoav Artzi, Victoria Basmova, Jonathan Berant, Ben Bogin, Sihao Chen, Pradeep Dasigi, Dheeru Dua, Yanai Elazar, Ananth Gottumukkala, et al. Evaluating models' local decision boundaries via contrast sets. *arXiv preprint arXiv:2004.02709*, 2020. 3

[18] Ayaan Haque, Matthew Tancik, Alexei A Efros, Aleksander Holynski, and Angjoo Kanazawa. Instruct-nerf2nerf: Editing 3d scenes with instructions. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 19740–19750, 2023. 3

[19] Yining Hong, Haoyu Zhen, Peihao Chen, Shuhong Zheng, Yilun Du, Zhenfang Chen, and Chuang Gan. 3d-llm: Injecting the 3d world into large language models. *Advances in Neural Information Processing Systems*, 36:20482–20494, 2023. 1, 2, 6, 8

[20] Wenbo Hu, Yining Hong, Yanjun Wang, Leison Gao, Zibu Wei, Xingcheng Yao, Nanyun Peng, Yonatan Bitton, Idan Szpektor, and Kai-Wei Chang. 3dllm-mem: Long-term spatial-temporal memory for embodied 3d large language model. *arXiv preprint arXiv:2505.22657*, 2025. 1

[21] Haifeng Huang, Yilun Chen, Zehan Wang, Rongjie Huang, Runsen Xu, Tai Wang, Luping Liu, Xize Cheng, Yang Zhao, Jiangmiao Pang, et al. Chat-scene: Bridging 3d scene and large language models with object identifiers. *Advances in Neural Information Processing Systems*, 37: 113991–114017, 2024. 1, 4, 6, 8, 2, 3

[22] Jiangyong Huang, Silong Yong, Xiaojian Ma, Xiongkun Linghu, Puhao Li, Yan Wang, Qing Li, Song-Chun Zhu, Baoxiong Jia, and Siyuan Huang. An embodied generalist agent in 3d world. *arXiv preprint arXiv:2311.12871*, 2023. 1, 2, 3, 8

[23] Jiangyong Huang, Baoxiong Jia, Yan Wang, Ziyu Zhu, Xiongkun Linghu, Qing Li, Song-Chun Zhu, and Siyuan Huang. Unveiling the mist over 3d vision-language understanding: Object-centric evaluation with chain-of-analysis. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 24570–24581, 2025. 1, 6, 8, 9

[24] Ting Huang, Zeyu Zhang, and Hao Tang. 3d-r1: Enhancing reasoning in 3d vlms for unified scene understanding. *arXiv preprint arXiv:2507.23478*, 2025. 1, 2, 8

[25] Divyansh Kaushik, Eduard Hovy, and Zachary C Lipton. Learning the difference that makes a difference with counterfactually-augmented data. *arXiv preprint arXiv:1909.12434*, 2019. 3

[26] Zaid Khan, Elias Stengel-Eskin, Jaemin Cho, and Mohit Bansal. Dataenvgym: Data generation agents in teacher environments with student feedback. *ArXiv*, abs/2410.06215, 2024. 3

[27] Chengen Lai, Shengli Song, Sitong Yan, and Guangneng Hu. Improving vision and language concepts understanding with multimodal counterfactual samples. In *European Conference on Computer Vision*, pages 174–191. Springer, 2024. 3

[28] Dong In Lee, Hyeongcheol Park, Jiyoung Seo, Eunbyung Park, Hyunje Park, Ha Dam Baek, Sangheon Shin, Sangmin Kim, and Sangpil Kim. Editsplat: Multi-view fusion and attention-guided optimization for view-consistent 3d scene editing with 3d gaussian splatting. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 11135–11145, 2025. 3

[29] Jialu Li, Hao Tan, and Mohit Bansal. Envedit: Environment editing for vision-and-language navigation. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15386–15396, 2022. 3

[30] Rong Li, Shijie Li, Lingdong Kong, Xulei Yang, and Junwei Liang. Seeground: See and ground for zero-shot open-vocabulary 3d visual grounding. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 3707–3717, 2025. 2

[31] Xiongkun Linghu, Jiangyong Huang, Xuesong Niu, Xiaojian Shawn Ma, Baoxiong Jia, and Siyuan Huang. Multi-modal situated reasoning in 3d scenes. *Advances in Neural Information Processing Systems*, 37:140903–140936, 2024. 1, 3

[32] Xiaojian Ma, Silong Yong, Zilong Zheng, Qing Li, Yitao Liang, Song-Chun Zhu, and Siyuan Huang. Sqa3d: Situated question answering in 3d scenes. *arXiv preprint arXiv:2210.07474*, 2022. 8, 2

[33] Xianzheng Ma, Yash Bhalgat, Brandon Smart, Shuai Chen, Xinghui Li, Jian Ding, Jindong Gu, Dave Zhenyu Chen, Songyou Peng, Jia-Wang Bian, et al. When llms step into the 3d world: A survey and meta-analysis of 3d tasks via multi-modal large language models. *arXiv preprint arXiv:2405.10255*, 2024. 1

[34] Rowan Hall Maudslay, Hila Gonen, Ryan Cotterell, and Simone Teufel. It's all in the name: mitigating gender bias with name-based counterfactual data substitution. *arXiv preprint arXiv:1909.00871*, 2019. 3

[35] Thomas Melistas, Nikos Spyrou, Nefeli Gkouti, Pedro Sanchez, Athanasios Vlontzos, Yannis Panagakis, Giorgos Papanastasiou, and Sotirios A Tsaftaris. Benchmarking counterfactual image generation. *Advances in Neural Information Processing Systems*, 37:133207–133230, 2024. 3

[36] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002. 2

[37] Amin Parvaneh, Ehsan Abbasnejad, Damien Teney, Javen Qinfeng Shi, and Anton Van den Hengel. Counterfactual vision-and-language navigation: Unravelling the unseen. *Advances in neural information processing systems*, 33:5296–5307, 2020. 3

[38] Axel Sauer and Andreas Geiger. Counterfactual generative networks. *arXiv preprint arXiv:2101.06046*, 2021. 3

[39] Jonas Schult, Francis Engelmann, Alexander Hermans, Or Litany, Siyu Tang, and Bastian Leibe. Mask3d: Mask

transformer for 3d semantic instance segmentation. *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8216–8223, 2022. 4

[40] Liang Song, Guangming Wang, Jiuming Liu, Zhenyang Fu, Yanzi Miao, et al. Sc-nerf: Self-correcting neural radiance field with sparse views. *arXiv preprint arXiv:2309.05028*, 2023. 3

[41] Damien Teney, Ehsan Abbasnedjad, and Anton van den Hengel. Learning what makes a difference from counterfactual examples and gradient supervision. In *European Conference on Computer Vision*, pages 580–599. Springer, 2020. 3

[42] Gladys Tyen, Hassan Mansoor, Victor Cărbune, Yuanzhu Peter Chen, and Tony Mak. Llms cannot find reasoning errors, but can correct them given the error location. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 13894–13908, 2024. 3

[43] Jason Uwaeze, Pranav Kulkarni, Vladimir Braverman, Michael A Jacobs, and Vishwa S Parekh. Generative counterfactual augmentation for bias mitigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1153–1160, 2025. 3

[44] Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575, 2015. 2

[45] Danqing Wang and Lei Li. Learning from mistakes via cooperative study assistant for large language models. *arXiv preprint arXiv:2305.13829*, 2023. 3

[46] Hanqing Wang, Wei Liang, Jianbing Shen, Luc Van Gool, and Wenguan Wang. Counterfactual cycle-consistent learning for instruction following and generation in vision-language navigation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15471–15481, 2022. 3

[47] Sijia Wang and Lifu Huang. Targeted augmentation for low-resource event extraction. *arXiv preprint arXiv:2405.08729*, 2024. 3

[48] Zehan Wang, Haifeng Huang, Yang Zhao, Ziang Zhang, and Zhou Zhao. Chat-3d: Data-efficiently tuning large language model for universal dialogue of 3d scenes. *arXiv preprint arXiv:2308.08769*, 2023. 1, 2, 6, 8

[49] Minghao Wen, Shengjie Wu, Kangkan Wang, and Dong Liang. Intergsedit: Interactive 3d gaussian splatting editing with 3d geometry-consistent attention prior. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 26136–26145, 2025. 3

[50] Runsen Xu, Xiaolong Wang, Tai Wang, Yilun Chen, Jiangmiao Pang, and Dahua Lin. Pointllm: Empowering large language models to understand point clouds. In *European Conference on Computer Vision*, pages 131–147. Springer, 2024. 1

[51] Wenda Xu, Daniel Deutsch, Mara Finkelstein, Juraj Juraska, Biao Zhang, Zhongtao Liu, William Yang Wang, Lei Li, and Markus Freitag. Llmrefine: Pinpointing and refining large language models via fine-grained actionable feedback. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 1429–1445, 2024. 3

[52] Ziyang Yan, Lei Li, Yihua Shao, Siyu Chen, Zongkai Wu, Jenq-Neng Hwang, Hao Zhao, and Fabio Remondino. 3dsceneeditor: Controllable 3d scene editing with gaussian splatting. *arXiv preprint arXiv:2412.01583*, 2024. 3

[53] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. 4

[54] Zeyuan Yang, Peng Li, and Yang Liu. Failures pave the way: Enhancing large language models through tuning-free rule accumulation. *arXiv preprint arXiv:2310.15746*, 2023. 3

[55] Barry Menglong Yao, Qifan Wang, and Lifu Huang. Error-driven data-efficient large multimodal model tuning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 20289–20306, 2025. 3

[56] Hanxun Yu, Wentong Li, Song Wang, Junbo Chen, and Jianke Zhu. Inst3d-lmm: Instance-aware 3d scene understanding with multi-modal instruction tuning. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 14147–14157, 2025. 2, 6

[57] Zhihao Yuan, Jinke Ren, Chun-Mei Feng, Hengshuang Zhao, Shuguang Cui, and Zhen Li. Visual programming for zero-shot open-vocabulary 3d visual grounding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20623–20633, 2024. 2

[58] Abhaysinh Zala, Jaemin Cho, Han Lin, Jaehong Yoon, and Mohit Bansal. Envgen: Generating and adapting environments via llms for training embodied agents. *ArXiv*, abs/2403.12014, 2024. 3

[59] Weichen Zhang, Ruiying Peng, Chen Gao, Jianjie Fang, Xin Zeng, Kaiyuan Li, Ziyou Wang, Jinqiang Cui, Xin Wang, Xinlei Chen, et al. The point, the vision and the text: Does point cloud boost spatial reasoning of large language models? *arXiv preprint arXiv:2504.04540*, 2025. 1

[60] Yiming Zhang, ZeMing Gong, and Angel X Chang. Multi3drefer: Grounding text description to multiple 3d objects. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15225–15236, 2023. 6, 7, 2

[61] Yue Zhang, Ziqiao Ma, Jialu Li, Yanyuan Qiao, Zun Wang, Joyce Chai, Qi Wu, Mohit Bansal, and Parisa Kordjamshidi. Vision-and-language navigation today and tomorrow: A survey in the era of foundation models. *arXiv preprint arXiv:2407.07035*, 2024. 1

[62] Yue Zhang, Zhiyang Xu, Ying Shen, Parisa Kordjamshidi, and Lifu Huang. Spartun3d: Situated spatial understanding of 3d world in large language model. In *The Thirteenth International Conference on Learning Representations*, 2025. 1, 2, 3

[63] Lichen Zhao, Daigang Cai, Lu Sheng, and Dong Xu. 3dvg-transformer: Relation modeling for visual grounding on point clouds. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2908–2917, 2021. 2

[64] Hongyan Zhi, Peihao Chen, Junyan Li, Shuailei Ma, Xinyu Sun, Tianhang Xiang, Yinjie Lei, Mingkui Tan, and Chuang Gan. Lscenellm: Enhancing large 3d scene understanding

using adaptive visual preferences. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 3761–3771, 2025. 2

[65] Ziyu Zhu, Xiaojian Ma, Yixin Chen, Zhidong Deng, Siyuan Huang, and Qing Li. 3d-vista: Pre-trained transformer for 3d vision and text alignment. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2911–2921, 2023. 6, 2

[66] Ziyu Zhu, Zhuofan Zhang, Xiaojian Ma, Xuesong Niu, Yixin Chen, Baoxiong Jia, Zhidong Deng, Siyuan Huang, and Qing Li. Unifying 3d vision-language understanding via promptable queries. In *European Conference on Computer Vision*, pages 188–206. Springer, 2024. 2, 6

[67] Jingyu Zhuang, Chen Wang, Liang Lin, Lingjie Liu, and Guanbin Li. Dreameditor: Text-driven 3d scene editing with neural fields. In *SIGGRAPH Asia 2023 Conference Papers*, pages 1–10, 2023. 3

[68] Ran Zmigrod, Sabrina J Mielke, Hanna Wallach, and Ryan Cotterell. Counterfactual data augmentation for mitigating gender stereotypes in languages with rich morphology. *arXiv preprint arXiv:1906.04571*, 2019. 3

Error-Driven Scene Editing for 3D Grounding in Large Language Models

Supplementary Material

A. More Details about Deer3D Framework

In this section, we provide additional implementation details about the DEER-3D framework. We elaborate on the specific prompts used for our Decomposition module (Sec. 3.2), the full statistical analysis of the instruction types that justifies our focus on Appearance and Spatial categories (Sec. 3.3), and the complete color palette $\mathcal{C}$ used for our Recolor operation (Sec. 3.4.1).

A.1. Prompt for Decompose Instruction

As detailed in Sec. 3.2 of the main paper, our DEER-3D framework begins by decomposing the full instruction into a set of general, atomic predicates. We use a large language model (e.g., Qwen3) for this task.To ensure the generated sub-instructions are consistent, unambiguous, and suitable for independent evaluation, we use a structured prompt. This prompt instructs the LLM to separate the instruction into a list of sentences, each focusing on a single semantic perspective (e.g., attribute, spatial relation). The prompt enforces several key constraints to maintain quality: it explicitly forbids the use of ambiguous referring expressions (like "it" or "them") and requires that all output sentences maintain the same, explicit subject (e.g., "The table is brown," "The table is near the wall"). The exact prompt used is shown below.

Prompt

```
Given an instruction sentence, separate
it into a list of sentences where each
focuses on only one perspective, such
as spatial relationship, attribute, or
quantity. Important constraints:
• Do NOT use referring expressions like
  "it" or "them". Explicitly state what
  these refer to.
• Ensure that all output sentences use
  the SAME subject throughout.
• If other objects are involved (e.g.,
  "a computer is placed on top of it"),
  rephrase so the original subject
  remains the focus (e.g., "The table
  has a computer placed on top of it").
Example:
Input: "There is a brown wooden table,
a computer is placed on top of it."
Output: ["There is a brown table",
"There is a wooden table", "The table
has a computer placed on top of it."]
```

A.2. Analysis of Error Distribution

In our methodology (Sec. 3.3), we state that our framework focuses on Appearance and Spatial errors. This decision is data-driven, based on a statistical analysis of the instruction types present in the training data. To quantify the most common semantic components, we randomly sample around 300 instructions from the training dataset. We then ran our Decomposition module (Sec. 3.2) on this sample to parse each instruction into its constituent atomic predicates (around 1500 sub-instruction). The distribution of predicate types within this representative sample is presented in Figure 5. The analysis reveals that the instructions are not uniformly distributed; they are overwhelmingly concentrated in two primary categories: Spatial Relations (39.6%) and Appearance (38.6%). Together, these two categories conclusively account for 78.2% of all semantic predicates found in our analysis. Assuming this large sample is representative of the full dataset, this provides clear empirical evidence that Spatial and Appearance are the most dominant semantic components in the data. Therefore, our DEER-3D framework is justifiably prioritized to focus its diagnostic and editing efforts on these two categories.

A.3. Edit Strategy for Recolor

In Sec. 3.4.1, our Recolor operation identifies the most perceptually distinct color c^* by maximizing its distance from a candidate set $\mathcal{C}$. To ensure that this process is both reproducible and perceptually meaningful, we define $\mathcal{C}$ as a predefined palette of 19 commonly used color prototypes. We operate in the CIELAB color space due to its perceptual uniformity, which makes Euclidean distances better aligned with human color perception. Our full LAB dictionary, and the candidate set $\mathcal{C}$ is listed below, mapping each color name to its corresponding (L*, a*, b*) values.

Table 5. The predefined set of color ($\mathcal{C}$) used for recolor operation.

Color Name	CIELAB (L, a, b)	Color Name	CIELAB (L, a, b)
white	(90, 0, 0)	green	(55, -35, 35)
black	(12, 0, 0)	turquoise	(70, -35, -10)
gray	(55, 0, 0)	blue	(40, 5, -55)
beige	(68, 5, 18)	dark blue	(30, 5, -45)
tan	(62, 10, 22)	purple	(45, 45, -25)
brown	(35, 18, 28)	pink	(70, 30, 10)
dark brown	(28, 14, 20)	violet	(50, 35, -35)
red	(45, 60, 30)	silver	(70, 0, 0)
orange	(60, 35, 50)	gold	(65, 5, 35)
yellow	(75, 5, 70)	—	—

Table 6. Performance comparision on MultiRefer. ZT: Zero-shot, ST: Single-task, MT: Multi-task, D: Distractor. Best results are bolded.

Method	ZT w/o D F1	ZT w/ D F1	ST w/o D F1@0.25	ST w/o D F1@0.5	ST w/ D F1@0.25	ST w/ D F1@0.5	MT F1@0.25	MT F1@0.5	Overall F1@0.25	Overall F1@0.5
3DVG-Trans+ [63]	87.1	45.8	–	27.5	–	16.7	–	26.5-	-	25.5
D3Net (Grounding) [7]	81.6	32.5	–	36.6	–	23.3	–	35.0	-	32.2
3DJCG (Grounding) [4]	**94.1**	66.9	–	26.0	–	16.7	–	26.2	-	26.6
M3DRef-CLIP [60]	81.8	39.4	53.5	47.8	34.6	30.6	43.6	37.9	42.8	38.4
Chat-Scene [21]	90.3	62.6	82.9	75.9	49.1	44.5	45.7	41.1	57.1	52.4
Ours	92.6	69.3	82.1	75.6	**52.6**	**48.4**	**50.0**	**46.0**	**60.0**	**55.8**

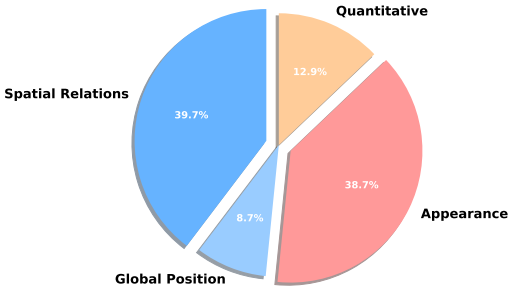


Figure 5. Distribution of semantic predicate types from our instruction analysis.

Table 7. Performance comparison on the ScanRefer.

Method	Unique		Multiple		Overall	
	Acc@0.25	Acc@0.5	Acc@0.25	Acc@0.5	Acc@0.25	Acc@0.5
3D-VisTA [65]	81.6	75.1	43.7	49.1	50.6	45.8
Chat-scene [21]	**89.6**	82.5	47.8	42.9	55.5	50.2
Ours	88.6	81.9	**50.8**	**45.8**	**57.8**	**52.3**

Table 8. Iterative refinement beyond Round 2 shows saturated performance.

Method	ScanRefer	
	Acc@0.25	Acc@0.5
Ours (Round 1)	57.8	55.8
Ours (Round 2)	**58.6**	**56.8**
Ours (Round 3)	58.6	56.7

B. More Details about Experiment

B.1. Implementation Details

Our training follows an iterative refinement paradigm. In Round 1, the model is fine-tuned on the initial batch of DEER-3D counterfactual scenes. The resulting model is then used to automatically identify new failure cases on the training set; these cases are subsequently edited to produce an expanded set of counterfactual samples for Round 2. This second-stage training further corrects the model's residual grounding errors. To improve stability during optimization, especially given the distribution shift introduced by edited scenes, we apply an Exponential Moving Average (EMA) over model parameters when training the final Round 2 model. EMA smoothing helps regularize updates, reduces variance in later training stages, and yields a more robust final checkpoint with improved grounding accuracy.

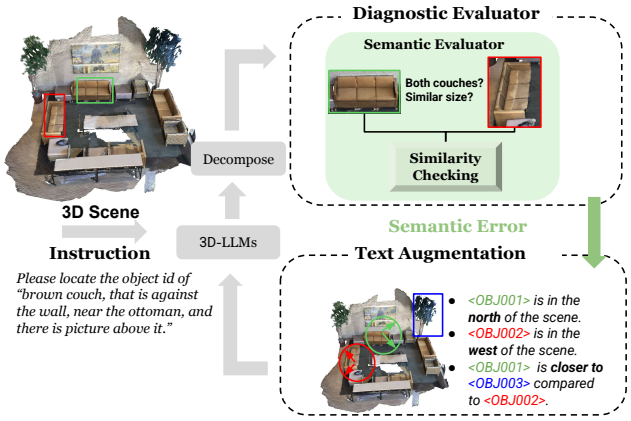


Figure 6. Semantic error detection followed by targeted text augmentation to disambiguate similar objects.

B.2. Performance on Scene Understanding Tasks

Beyond 3D grounding, we further assess our models on a suite of general scene understanding tasks, including Scan2Cap [11], ScanQA [3], and SQA3D [32]. For Scan2Cap, we evaluate captioning quality using CIDEr@0.5 (C@0.5) and BLEU-4@0.5 (B-4@0.5), following the standard practice in 3D captioning benchmarks. For ScanQA, we adopt the official evaluation metrics of CIDEr (C) [44] and BLEU-4 (B-4) [36] to measure answer relevance and linguistic fidelity. Finally, for SQA3D, we report Exact Match accuracy (EM) and its refined variant EM-R [22], which provide a more granular assessment of spatially grounded question answering. Our method achieves consistent improvements over the baseline across all benchmarks, indicating that the benefits of our approach extend beyond grounding-specific settings.

B.3. More Results on Grounding Task

In the main paper, we report only the overall grounding performance on ScanRefer and Multi3DRefer. For completeness, we provide per–subset results for these benchmarks in Table 6 and Table 7, respectively.

To further investigate whether iterative refinement continues to improve performance beyond Round 2, we perform an additional round of error mining and fine-tuning using the Round 2 model as the starting point. As shown in Table 8, the resulting Round 3 model exhibits almost no

Table 9. Performance comparison on 3D grounding benchmarks.

Method	ScanRefer	
	Acc@0.25	Acc@0.5
Chat-Scene [21]	55.5	50.2
Random-Text	56.1	50.3
DEER-3D+Text-Aug	**56.6**	**50.8**

improvement over Round 2 (differences within ± 0.1 across metrics). This suggests that the model has already corrected the major grounding failures by the second iteration, and that additional refinement yields diminishing returns. These results confirm our claim in the main paper that Round 2 represents a practical stopping point that achieves the best balance between accuracy gains and computational cost.

B.4. Error-Driven Framework Evaluation

Beyond generating targeted 3D scene edits, our error-driven framework can also produce targeted text augmentations that disambiguate instructions. As shown in Fig. 6, we design an experiment focusing on a subset of grounding failures where the predicted and ground-truth objects belong to the same semantic class and exhibit similar sizes and appearances. These failures typically arise from instructional ambiguity, where the textual description does not provide sufficient cues for the model to identify the correct instance.

Our diagnostic evaluator detects such cases by checking semantic consistency between the predicted and ground-truth objects (e.g., class match and size similarity). When an error is attributed to ambiguous language rather than visual misinterpretation, the framework triggers the text augmentation module instead of applying a 3D edit. This module generates precise disambiguating descriptions by incorporating contextual object references that uniquely characterize the target, such as: (1) global position cues (e.g., "located in the west of the scene"), (2) local relational attributes (e.g., "closer to <OBJ003> than <OBJ002>").

These augmentations enrich the instruction with predicate-level distinctions, enabling the model to differentiate between visually similar candidates without modifying the underlying scene. As shown in Table 9, simple random text perturbations yield minimal improvement, indicating that generic augmentation is insufficient for resolving ambiguity. In contrast, our targeted text augmentation provides a better performance, demonstrating that error-driven, semantically grounded clarifications improve disambiguation. However, the overall gains remain relatively modest compared to our full editing pipeline, suggesting that text-only augmentation is helpful but inherently limited for 3D grounding tasks.

C. Limitation and Future Work

Despite the improvements introduced by DEER-3D, several broader challenges in current 3D-LLMs remain. Many grounding failures originate from limitations inherent to point-cloud–based 3D perception pipelines, such as incomplete geometry or reconstruction noise. The capability of the base 3D-LLM also fundamentally constrains the upper bound of achievable performance, as weaker reasoning or recognition ability limits the effectiveness of error diagnosis and correction. In addition, our current framework operates on datasets of relatively modest scale and primarily focuses on indoor environments, which restricts the magnitude and generality of performance gains. Finally, although our counterfactual edits introduce targeted and meaningful variations, they still cannot cover the long-tail complexity of real-world 3D scenes involving deformable objects, human interactions, or dynamic environments. Future work may explore scaling DEER-3D to larger and more diverse 3D corpora, leveraging stronger 3D-LLMs for better perception and reasoning, and enriching the editing module to support more complex and varied scenes.