



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

DOTTORATO DI RICERCA IN
DATA SCIENCE AND COMPUTATION

Ciclo 37

Settore Concorsuale: 09/H1 - SISTEMI DI ELABORAZIONE DELLE INFORMAZIONI

Settore Scientifico Disciplinare: ING-INF/05 - SISTEMI DI ELABORAZIONE DELLE INFORMAZIONI

MULTIMODAL AI FOR 3D-LANGUAGE UNDERSTANDING

Presentata da: Andrea Amaduzzi

Coordinatore Dottorato

Daniele Bonacorsi

Supervisore

Luigi Di Stefano

Co-supervisore

Giuseppe Lisanti

Esame finale anno 2026

Declaration of Authorship

I, Andrea AMADUZZI, declare that this Thesis titled, “Multimodal AI for 3D-language Understanding” and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this Thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this Thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the Thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Andrea Amaduzzi

Date:

February 10, 2026

*“Non si possono trovare verità e creatività in un terremoto,
solo in una ricerca silenziosa.”*

Umberto Eco

Abstract

Understanding the world through natural language is a fundamental challenge in computer vision, requiring models to bridge the gap between geometric representations and semantic understanding. While recent advances in deep learning have enabled impressive progress in multimodal tasks, effectively connecting 3D data with language remains difficult due to the computational complexity of 3D representations and the scarcity of high-quality paired 3D-language datasets. In this Thesis, we focus on developing novel deep learning techniques that can efficiently process and describe 3D data through natural language. First, we address the critical lack of high quality 3D-language datasets and reliable evaluation methods for text-to-3D generation by introducing a benchmark comprising GPT2Shape, a dataset of 3D models with high-quality language annotations, CrossCoherence, a novel quantitative metric for evaluating text-to-3D coherence, and HST, a human-validated test set. This benchmark demonstrates superior performance compared to existing 2D-based evaluation metrics. In the second part of this Thesis, we introduce NeRF-language models — a new paradigm of Multimodal Large Language Models that directly process the weights of Neural Radiance Fields (NeRFs) rather than explicit representations like images or point clouds materialized from them. We show that this approach enables holistic 3D understanding without computationally expensive rendering, achieving state-of-the-art performance on the first NeRF-language datasets. Finally, we propose a method for computing spatially-aware tokens directly from the weights of a NeRF, enabling fine-grained spatial understanding and detailed scene comprehension. Our resulting model demonstrates strong capabilities in spatial reasoning on multi-object scenes, with promising generalization to real-world scenarios.

Contents

Declaration of Authorship	iii
Abstract	v
1 Introduction	1
1.1 3D data and language	1
1.1.1 3D representations	1
1.1.2 Brief history of Research on 3D-language	3
1.2 Thesis Organization	5
2 Looking at words and points with attention	7
2.1 Introduction	7
2.2 Related Work	9
2.3 GPT2Shape	11
2.4 GPT2Shape vs Text2Shape: a user study	14
2.5 Human-validated Shape-Text dataset	16
2.6 <i>CrossCoherence</i>	16
2.6.1 Architecture and Training Objective	17
2.6.2 Training	18
2.7 Experimental Results	19
2.7.1 Comparison with existing metrics	19
2.7.2 Text-to-shape coherence of generative methods	22
2.8 Ablation experiments on the attention layers of <i>CrossCoherence</i>	25
2.9 Sensitivity of CLIP-based metrics to different 3D representations	26
2.10 Sensitivity of CLIP-based metrics to CLIP base model and number of renderings.	27
2.11 Conclusion	28
3 The Rise of NeRF-language Models	29
3.1 Introduction	29
3.2 Related Work	31
3.3 Methodology	34

3.3.1	Neural Radiance Fields	34
3.3.2	Meta-encoder	35
3.3.3	Large Language and NeRF Assistant	37
3.3.4	Training protocol	37
3.4	Benchmark	38
3.4.1	ShapeNeRF-Text	38
3.4.2	ObjaNeRF-Text	40
3.4.3	Language tasks and metrics	40
3.5	Experimental results	41
3.5.1	Experiments on ShapeNeRF-Text and ObjaNeRF-Text	41
3.5.2	Ablation study on training data	49
3.5.3	Is the LLM all you need?	50
3.5.4	Additional qualitative examples	52
3.6	Conclusion	54
4	Spatially-aware Weights Tokenization for NeRF-Language Models	55
4.1	Introduction	55
4.2	Related work	57
4.3	weights2space: spatially-aware NeRF weights tokenization	58
4.3.1	Architecture of weights2space	58
4.3.2	Design Rationale: Weights-to-Tri-plane or Tri-plane-to-Language	61
4.4	<i>Spatial LLaNA</i> : spatially-aware MLLM for NeRF	62
4.5	Experimental setup	63
4.6	Experimental results	65
4.6.1	Results on ShapeNeRF-Text, HST, and ObjaNeRF-Text.	65
4.6.2	Results on Spatial ObjaNeRF	68
4.6.3	Failure cases of <i>Spatial LLaNA</i>	70
4.6.4	Ablation on LLM finetuning	71
4.6.5	Real-world generalization of <i>Spatial LLaNA</i>	74
5	Final remarks	76
5.1	Summary of Thesis Achievements	76
5.2	Current limitations	77
5.3	Future Directions	77
	Bibliography	80

List of Figures

2.1	Benchmarking text-to-shape generative models calls for better datasets and better metrics. Top: as existing datasets [20] contain many uninformative descriptions (left), we automatically create high-quality text prompts (right) by leveraging GPT-3 [15]. Bottom: the existing metric CLIP-Similarity judges the left shape as more coherent to the given text than the right one, whilst our novel <i>CrossCoherence</i> prefers the right shape over the left one.	8
2.2	An example of the text prompts rephrasing process. The green box contains Text2Shape descriptions for a shape together with the designed request; their combination represents the input prompt to GPT-3. The blue box is the output of GPT-3, providing the textual descriptions of GPT2Shape.	12
2.3	Example of Text2Shape and GPT2Shape textual annotations. The green box contains the original Text2Shape descriptions for the 3D shape in the middle. The blue box shows the rephrased text prompts of GPT2Shape. In yellow, we highlight the ambiguous or wrong descriptions of Text2Shape.	13
2.4	Layout of the user study	14
2.5	Architecture of the PointNet++ autoencoder. In red the features used by <i>CrossCoherence</i> , and in blue the embeddings that have been used to compute the <i>easy</i> and <i>hard</i> distractors for each reference shape.	15
2.6	Architecture of <i>CrossCoherence</i>	17
2.7	Qualitative results of CLIP-Similarity and <i>CrossCoherence</i> on HST dataset. The green check indicates the shape associated with the prompt, while the red cross identifies the distractor. For all these triplets, CLIP-Similarity prefers the left shape while <i>CrossCoherence</i> the right one.	21

2.8	Qualitative results of CLIP R-precision and <i>CrossCoherence</i> on HST dataset. The green check indicates the ground truth text. For all these triplets, CLIP R-precision retrieves the left text while <i>CrossCoherence</i> the right one.	21
2.9	Examples of shapes generated by Point-E (left), Shap-E (center) and Liu et al. (right) for the given text prompt. As highlighted, for all these triplets, CLIP-Similarity prefers the shape in the middle while <i>CrossCoherence</i> the one on the right.	23
2.10	Examples of shapes generated by Shap-E with the corresponding text retrieved by CLIP R-precision, on the right, and <i>CrossCoherence</i> , on the left.	24
3.1	LLaNA. A new Multimodal Large Language Model that understands and reasons on an input NeRF. Notably, our framework processes directly the NeRF weights and performs tasks such as captioning, Q&A, and zero-shot classification of NeRFs. . .	31
3.2	Architecture of nf2vec.	36
3.3	Framework overview. Example of NeRF captioning. The meta-encoder of this model is nf2vec, shown in Figure 3.2 . .	36
3.4	Automatic annotation pipeline	38
3.5	Example of data sample from ShapeNeRF-Text	39
3.6	Qualitative results on ShapeNeRF-Text brief descriptions. . .	43
3.7	Qualitative results on ObjaNeRF-Text brief descriptions (PointLLM test set).	44
3.8	Qualitative results on ObjaNeRF-Text brief descriptions (GPT4Point test set).	44
3.9	Qualitative results on ShapeNeRF-Text detailed descriptions. From top to bottom: brief and detailed descriptions, single-round Q&A	45
3.10	Qualitative results on ShapeNeRF-Text single-round Q&A.	46
3.11	Brief description example from ShapeNeRF-Text.	52
3.12	Detailed captioning example from ShapeNeRF-Text.	53
3.13	Single-round Q&A example from ShapeNeRF-Text.	53

4.1	(left) LLaNA relies on a single global representation of the input NeRF. (right) <i>S-LLaNA</i> processes multiple spatially-aware tokens extracted by our new meta-encoder and, thus, can provide detailed descriptions that capture the relationships between objects. In particular, the answer of <i>S-LLaNA</i> contains more fine-grained information (e.g., it hints at the “ <i>wooden shelf with a red ceramic pot</i> ” and enumerates the “ <i>four wheels</i> ” and the “ <i>three wooden benches</i> ”) while also reasoning spatially (e.g., it discriminates between objects “ <i>on one side</i> ” and objects “ <i>Inside the carriage</i> ”).	56
4.2	Overview of weights2space.	58
4.3	Overview of <i>Spatial LLaNA</i>.	61
4.4	Comparison between the annotations of <i>ObjaNeRF-Text</i> (GPT4Point test set) and <i>Spatial ObjaNeRF</i> . In green , the fine-grained spatial details.	65
4.5	Qualitative evaluation of the brief captioning task on <i>ShapeNeRF-Text</i> and <i>ObjaNeRF-Text</i> . In red , the hallucinations generated by the MLLMs. In green , the correct fine-grained and spatial details.	67
4.6	Qualitative evaluation of the detailed captioning task on <i>ShapeNeRF-Text</i> . In red , the hallucinations generated by the MLLMs. In green , the correct fine-grained and spatial details.	68
4.7	Qualitative evaluation of the single-round Q&A task on <i>ShapeNeRF-Text</i> . In red , the hallucinations generated by the MLLMs. In green , the correct fine-grained and spatial details.	68
4.8	Qualitative examples on <i>Spatial ObjaNeRF</i>	69
4.9	Qualitative evaluation on <i>Spatial ObjaNeRF</i> . In red , the hallucinations generated by the MLLMs. In green , the correct fine-grained and spatial details.	70
4.10	Examples of failure cases of <i>Spatial LLaNA</i> on <i>Spatial ObjaNeRF</i> . In red , the hallucinations generated by the model. In green , the correct fine-grained and spatial details.	71
4.11	Visualization of the self-attention maps of the LLM (LLaMA 2)	73
4.12	Visualization of the normalized self-attention scores of single NeRF tokens towards the text tokens of the question. In yellow, surrounded by a red circle, the reference NeRF token.	73
4.13	Qualitative examples of <i>S-LLaNA</i> on <i>OmniObject3D</i>	75

4.14 Qualitative examples of <i>S-LLaNA</i> on Mip-NeRF 360	75
---	----

List of Tables

2.1	Summary of user study results	16
2.2	Comparison between ShapeGlot, <i>CrossCoherence</i> and CLIP-Similarity. The first column reports accuracy for methods trained and tested only on chairs, while the second column is for methods trained and tested on the full HST.	19
2.3	Comparison between ShapeGlot, <i>CrossCoherence</i> and CLIP R-precision based on the R-precision protocol. The first row reports the results for methods trained and tested only on chairs, while the second row is for methods trained and tested on full HST.	19
2.4	Comparison of <i>CrossCoherence</i> and CLIP R-precision across generative models. Each entry in a column reports how often a metric considers a prompt more coherent with the shape generated by one method versus the others.	22
2.5	Comparison of <i>CrossCoherence</i> and CLIP R-precision across generative models.	23
2.6	Comparison on the HST test set between <i>CrossCoherence</i> using different attention schemes. The second column specifies the kind of attention used in the model. The third column reports accuracy for models trained and tested only on chairs, while the fourth column shows the results for models trained and tested on both chairs and tables.	26
2.7	Sensitivity study of CLIP-similarity to different 3D representations.	27
2.8	Sensitivity study of CLIP R-precision to different 3D representations.	27
2.9	Performance of CLIP-Similarity on HST dataset using different CLIP base models and a different number of rendered views.	27
3.1	NeRF brief captioning results on ShapeNeRF-Text. Best results are in bold , runner-up is <u>underlined</u>	42

3.2	NeRF brief captioning results on HST. Best results are in bold , runner-up is <u>underlined</u>	42
3.3	NeRF brief captioning results on ObjaNeRF-Text (PointLLM test set). Best results are in bold , runner-up is <u>underlined</u> . . .	42
3.4	NeRF brief captioning results on ObjaNeRF-Text (GPT4Point	43
3.5	NeRF detailed captioning on ShapeNeRF-Text. Best results are in bold , runner-up is <u>underlined</u> . (FV: front-view, BV: back-view, MV: multi-view)	45
3.6	NeRF single-round Q&A on ShapeNeRF-Text . Best results are in bold , runner-up is <u>underlined</u> . (FV: front-view, BV: back-view, MV: multi-view)	47
3.7	Zero-Shot NeRF Classification . Best results are in bold , runner-up is <u>underlined</u> . (FV: front-view, BV: back-view, MV: multi-view)	48
3.8	Brief captioning on ShapeNeRF-Text of trained baselines . Best results are in bold , runner-up is <u>underlined</u>	49
3.9	Brief captioning on HST of trained baselines . Best results are in bold , runner-up is <u>underlined</u>	49
3.10	Detailed captioning on ShapeNeRF-Text of trained baselines . Best results are in bold , runner-up is <u>underlined</u>	49
3.11	Single-round Q&A on ShapeNeRF-Text of trained baselines . Best results are in bold , runner-up is <u>underlined</u>	49
3.12	Zero-shot NeRF classification of baselines trained on ShapeNeRF-Text . Best results are in bold , runner-up is <u>underlined</u> . (FV: front-view, MV: multi-view)	50
3.13	Language-only baselines NeRF captioning and NeRF single-round Q&A . (Shape: ShapeNeRF-Text, Obja-P: ObjaNeRF-Text PointLLM test set, Obja-G: ObjaNeRF-Text GPT4Point test set)	50
4.1	NeRF brief captioning on ShapeNeRF-Text and HST datasets . Best results are in bold , runner-up is <u>underlined</u> . (FV: front-view, BV: back-view, MV: multi-view)	65
4.2	NeRF brief captioning on ObjaNeRF-Text . Best results are in bold , runner-up is <u>underlined</u> . (RV: random view, MV: multi-view)	66
4.3	NeRF detailed captioning on ShapeNeRF-Text	66
4.4	NeRF single-round Q&A on ShapeNeRF-Text	66
4.5	Detailed spatial captioning on Spatial ObjaNeRF . Best results are in bold , runner-up is <u>underlined</u> . (RV: random view, MV: multi-view)	68

4.6	Experiments on LLM finetuning. Best results are in bold , runner-up is <u>underlined</u> . (SN-T: ShapeNeRF-Text, ON-T: ObjaNeRF-Text PointLLM test set)	71
-----	---	----

List of Abbreviations

LLM	Large Language Model
VLM	Vision-Language Model
MLLM	Multimodal Large Language Model
NeRF	Neural Radiance Field
MLP	Multi Layer Perceptron
3DGS	3D Gaussian Splatting

Chapter 1

Introduction

Thanks to the transformative development of deep learning, many applications connecting visual concepts and text have been developed. CLIP [99] has been a pioneering and influential work in this direction, introducing a framework to learn joint representations of images and natural language. This paradigm has reshaped research in multimodal learning, inspiring numerous applications across vision, language, and their intersection. In this field, numerous works have extended the paradigm, enabling applications like image captioning [115], generation [140] and visual-question answering [67].

While most progress so far has focused on visual data, the natural next step of this research field was to connect language and three-dimensional data, which offer richer representations of the physical world. Unlocking this connection opens the door to powerful opportunities in domains such as robotics, augmented reality, and digital content creation, where reasoning about spatial structure and geometry is essential. This Thesis aims to contribute to the development of this field of research, that establishes a common ground between 3D representations and natural language.

1.1 3D data and language

While language is a type of data with which, as humans, we are naturally familiar, it is useful to provide an overview of the main 3D representations employed in the research domain of this dissertation.

1.1.1 3D representations

Point clouds [61] represent 3D shapes as unordered set of points in 3D space, each with Cartesian coordinates, and optionally attributes like color or normal.

Voxel grids [77] consist of unit cubes in 3D space, analogous to pixel representations in 2D.

Polygonal mesh representations [13] are composed of vertices and surfaces, providing a compact description of complex 3D shapes.

Neural fields [120] have attracted increasing attention in the 3D research community over recent years. A neural field defines a mapping from spatial coordinates to scene properties (such as occupancy, color, radiance, etc). Unlike voxel grids, where the mapping associates each discrete cell with a stored value, neural fields rely on a learned function, typically a multi-layer perceptron. This enables them to implicitly capture compact, continuous, and differentiable representations of 3D shapes and scenes. One family of neural fields is known as Neural Radiance Fields (NeRFs) [88], which are designed to model and render view-dependent color and volume density (radiance) by querying a multi-layer perceptron with both spatial coordinates and viewing direction.

3D Gaussian Splatting [53] can be considered a variation of point cloud representations, where each point carries additional information describing the radiance emitted in the spatial neighborhood, modeled as an anisotropic 3D Gaussian “blob”. These Gaussians are typically initialized from structure-from-motion (SfM) point clouds and then optimized via differentiable rendering. By relying on efficient rasterization instead of ray tracing, 3D Gaussian Splattings enable state-of-the-art novel view synthesis.

Each of these representations presents distinct advantages and limitations, making each of them more suitable for certain applications than for others. For instance, **point clouds** are quite simple to acquire, especially with modern depth sensors and LiDAR devices, and are widely used in robotics and autonomous driving. However, their lack of explicit connectivity makes them less suitable for applications requiring detailed surface reconstructions, such as high-fidelity 3D modeling or physical simulations. **Meshes** are a compact and widely adopted format in graphics and computer-aided design, well-suited for rendering and animation pipelines. Nonetheless, their irregular structure and non-differentiability hinder their direct use in modern deep learning frameworks, unless combined with specialized processing techniques. **Voxel grids**, by contrast, provide a uniform and structured representation, which facilitates the use of convolutional neural networks for tasks such as 3D semantic segmentation. Yet, their cubic scaling in memory with resolution severely limits their practicality for large-scale or high-resolution scenes. **Neural fields** offer a powerful alternative by implicitly encoding continuous geometry and appearance through neural networks. This allows for differentiable pipelines and has led to breakthroughs in tasks such as

novel-view synthesis and surface reconstruction. Still, their training can be computationally intensive, and generalization across scenes often remains a challenge. Finally, **3D Gaussian Splatting** strikes a balance between efficiency and quality: it achieves state-of-the-art rendering performance with lower computational cost compared to NeRF, making it attractive for real-time applications. Nevertheless, its reliance on initialization from point clouds and the need for careful parameter tuning can limit its flexibility in scenarios where input data is sparse or noisy.

In summary, the choice of representation strongly depends on the target application: robotics and autonomous systems often rely on point clouds for their efficiency and sensor compatibility, graphics and CAD systems benefit from meshes for precise modeling and rendering, while neural fields and 3D Gaussian Splatting open new possibilities for photorealistic rendering and differentiable scene understanding.

1.1.2 Brief history of Research on 3D-language

The emergence of research in this area was inspired by the rapid development of the 2D-language field, which found one of its most prosperous applications in text-to-image generation [101, 106, 105]. Such models were enabled by the abundance of large-scale image-text datasets [108, 72, 113]. In contrast, research on 3D-language has faced a major obstacle from the very beginning: the lack of paired data.

One of the earliest applications in this field was text-to-3D shape generation. Text2Shape [20], the first work in this direction, introduced a method to generate 3D voxels from textual descriptions. The model was trained on a manually annotated dataset comprising only two object categories (chairs and tables) from ShapeNet [17], for a total of 15K objects. Faced with such scarcity of data, many works leveraged advances from the vision-language domain, including both representation learning models like CLIP [99] and text-to-image generative models [105]. This insight led to the development of numerous text-to-3D generation methods across different kinds of 3D representations: CLIP-Forge [107] for voxels, Liu et al. [76], Shap-E [52] and [36] for implicit functions, DreamFields [50], DreamFusion [95], Magic3D [70], and ProlificDreamer [130] for NeRFs, Point-E [92] for point clouds, TextMesh [123] for meshes, and GSGEN [24] and DreamGaussian [119] for Gaussian Splatting. In parallel, text-guided editing of 3D shapes was also proposed, enabling the

modification of style [86, 26], texture [104, 68], and geometry [42, 55] of 3D meshes.

The advent of Large Language Models (LLMs) has had a profound impact on 3D-language research. The remarkable reasoning and conversational abilities of models like LLaMA [122] and GPT-4 [1] stimulated progress in several directions. First, they led to the rise of Multimodal Large Language Models (MLLMs). These models are capable of performing language tasks using input data from diverse modalities, including images, videos, audio, and 3D. The earliest MLLMs to demonstrate strong potential were those for 2D vision-language tasks, fueled by the availability of massive image-text datasets. BLIP-2 [63] and LLaVA [74] achieved significant advancements in image comprehension and captioning. Crucially, these models enabled large-scale automatic generation of textual annotations for 3D datasets, producing collections with hundreds of thousands of 3D-language pairs [80, 133, 98]. The availability of such datasets provided new momentum for 3D-language research.

Therefore, MLLMs were extended to the 3D domain. This is the case for PointLLM [133], GPT4Point [133] and ShapeLLM [139], which all take single-object point clouds as input. The growing availability of large and diverse 3D-language datasets has further enabled the extension of 3D MLLMs to large indoor and outdoor scenes [43, 132, 73, 84].

More recently, a new line of research has emerged, leveraging Transformer-based models to generate 3D shapes autoregressively, in the same way that LLMs generate text. Several publications [134, 23, 21, 129, 112] have proposed models that, with one unified architecture, are capable of producing textual descriptions and dialogues as well as 3D assets in mesh form, by converting 3D representations into sequences of discrete tokens.

This Thesis situates itself within this literature, contributing along several directions. First, it addresses the construction of high-quality 3D-language datasets and the development of a dedicated metric for 3D-language alignment. Since most existing text-to-3D generation and editing models rely on 2D quantitative metrics, computed on images rendered from a 3D object — such as CLIP Similarity or CLIP R-Precision [99] — one contribution of this dissertation is to directly tackle this limitation. Specifically, we improve the quality of an existing 3D-language dataset using an LLM and propose the first metric designed to quantify the consistency between a colored point cloud and a textual description. This work is detailed in **Chapter 2**.

The dissertation also advances the research area of 3D MLLMs, introducing

the first MLLM designed for NeRFs, which until then had only been developed for explicit 3D data representations like point clouds and meshes. A peculiar feature of this work is that the proposed model can directly process the weights of the neural network that represents the NeRF of an object. Building upon this contribution, we further investigate and extend the capabilities of our model to more complex data and larger, more powerful LLMs. The details of this research are described in [Chapter 3](#).

Finally, we introduce a novel method for converting NeRF weights into a sequence of spatially-aware tokens, enabling the construction of a new MLLM for NeRFs with strong spatial reasoning capabilities. This contribution is presented in [Chapter 4](#).

1.2 Thesis Organization

The structure of this dissertation is as follows:

[Chapter 2](#) - *Looking at words and points with attention: a benchmark for text-to-shape coherence.*

In this chapter, we propose a benchmark for text-to-shape coherence, comprising a dataset of 3D assets paired with high-quality language annotations, a state-of-the-art quantitative metric employable on colored point clouds, and a human-validated test set, used to evaluate and compare text-driven 3D generative models.

[Chapter 3](#) - *The rise of NeRF-language models.* We introduce LLaNA, the first Multimodal Large Language Model (MLLM) for NeRFs, capable of performing new tasks such as NeRF captioning and Q&A. Notably, this method directly processes the weights of a NeRF’s MLP to extract information about the represented objects without the need to render images or materialize 3D data structures. The results of this work show that LLaNA performs favourably against MLLMs processing explicit 2D or 3D representations from NeRFs. We also propose an extension of LLaNA, in terms of data, model scale and experimental evaluation. First of all, we introduce *ObjaNeRF-Text*, the first large-scale dataset of NeRFs paired with language annotations, comprising 280K data samples. Secondly, we scale our model, by integrating a larger underlying LLM. Finally, we extend the previous experimental setup through a detailed scaling analysis of the model and an ablation study on the impact of the size of the LLM in state-of-the-art MLLMs on NeRF-language tasks.

[Chapter 4](#) - *Spatially-aware Weights Tokenization for NeRF-Language Models.* We address the problem of performing accurate and spatially-aware

NeRF-language tasks. To this end, we introduce *Spatial LLaNA*: a novel MLLM for NeRF, that relies on a new NeRF weights encoder. This encoder, `weights2space`, is able to compute a sequence of spatially-aware tokens from the weights of a NeRF’s MLP. This local representation is crucial to obtain excellent performance in NeRF-language tasks, allowing *Spatial LLaNA* to establish itself as the state-of-the-art MLLM for NeRFs, particularly effective on spatial reasoning.

Chapter 2

Looking at words and points with attention

This Chapter refers to the work *"Looking at words and points with attention: a benchmark for text-to-shape coherence"*, published at the peer-reviewed workshop "AI for 3D Content Creation" at the International Conference of Computer Vision (ICCV) 2023.

2.1 Introduction

Text-conditioned 3D object generation and manipulation has rapidly advanced in recent years, driven by progress in large generative models. However, the initial development of this field has been severely limited by two aspects. First, the lack of general, high-quality 3D-text datasets: at the time of the publication this Chapter refers to [5], the only datasets for text-driven shape generation were Text2Shape [20] and its extended version Text2Shape++ [36], containing text descriptions for chairs and tables from ShapeNet [17]. However, as shown in Figure 2.1 (top, left), the textual descriptions provided by these datasets are often generic and fail to capture all the key, fine-grained details of the objects in terms of both geometry and appearance. The lack of datasets with fine-grained descriptions poses a significant challenge in training effective generative models. A supervision that comes from imprecise and noisy text prompts hinders the ability of the model to generate high-quality and contextually accurate 3D data. Another important limitation is the lack of a common benchmark. It is impossible to overstate the importance of benchmarks in enabling and driving the rapid developments of deep learning: this is a cornerstone of the field since ImageNet [31] gave rise to the AlexNet breakthrough [59]. The definition of a clear benchmark for text-to-shape generation is hindered not only by the absence of high-quality paired 3D-text datasets but also by metrics. Most currently adopted quantitative metrics can measure only the generative

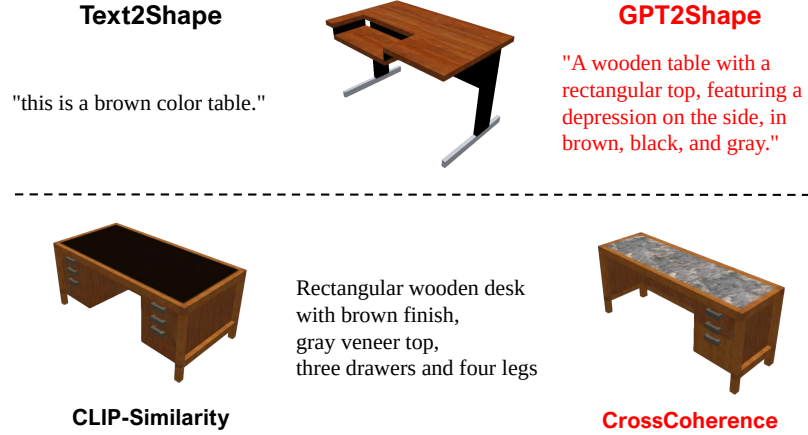


FIGURE 2.1: Benchmarking text-to-shape generative models calls for better datasets and better metrics. **Top:** as existing datasets [20] contain many uninformative descriptions (left), we automatically create high-quality text prompts (right) by leveraging GPT-3 [15]. **Bottom:** the existing metric CLIP-Similarity judges the left shape as more coherent to the given text than the right one, whilst our novel *CrossCoherence* prefers the right shape over the left one.

ability of a method, i.e., the realism of the generated shapes in terms of the distance between their distribution and the ground truth one. These metrics, however, are not able to quantify how closely a 3D shape fits a text, arguably the most important aspect to properly evaluate text-to-shape methods. Exceptions are metrics based on CLIP embeddings [99], which are computed from 2D renderings of the 3D shape and text embeddings. However, these approaches may turn out sensitive to rendering parameters and struggle to capture coherence with individual words or fine-grained geometric details due to the global nature of the CLIP text and visual embeddings. In this work, we propose the first benchmark for text-to-shape generation and manipulation that addresses both shortcomings highlighted above. Since collecting a novel dataset from scratch is a time-consuming and costly effort, we investigate the effectiveness of pre-trained large language models (LLMs) to leverage existing datasets and improve their quality at a fraction of the cost. In particular, as shown in Figure 2.1 (top, right), we create an improved version of Text2Shape, which we dub GPT2Shape, whose textual descriptions have been generated by the large language model GPT-3 [15] starting from the text prompts of the original dataset. The higher quality of such text prompts has been validated through a human evaluation study. **GPT2Shape** provides multiple fine-grained textual descriptions for every 3D shape. The detail and accuracy of textual descriptions can enable training of generative models, which can effectively capture and learn relationships between the words in the input

text and the localized details of the corresponding 3D shape. Such capability would be very hard to achieve with noisy and imprecise text prompts. We also propose a novel metric, dubbed *CrossCoherence*, which exploits the cross-attention mechanism to quantify the coherence between a colored point cloud and a textual description at the word and geometric detail level. We use the text-shape pairs validated through the human study to compare the existing text-to-shape coherence metrics quantitatively and show how *CrossCoherence* outperforms previous proposals (Figure 2.1, bottom). Our metric can also be used in a comparative scenario, where a text prompt and two corresponding point clouds are provided. *CrossCoherence* can determine which point cloud is more faithful to the text. Thus, the performance of different text-driven shape generation methods can be compared in terms of text fidelity. Finally, as a by-product of our human evaluation study, we created the **Human-validated Shape-Text (HST) dataset** by collecting the textual descriptions that have been coherently associated with a shape by the participants. The usefulness of the HST dataset is twofold: it provides a set of prompts and associated shapes, that can be used as a test set of our benchmark since they are not present in the training set but come from the same distribution on which *CrossCoherence* has been trained; it may also serve as a benchmark for the development of new text-to-shape coherence metrics.

To sum up, the main contributions described in this Chapter are:

- **GPT2Shape**, an automatically improved dataset consisting of shape-text pairs with high-quality, fine-grained textual descriptions;
- *CrossCoherence*, a state-of-the-art quantitative metric for text-to-shape coherence, which can be directly applied to RGB point clouds;
- **HST**, a human-validated test set where *CrossCoherence* can be used to evaluate and compare text-to-shape generative models and foster their development as well as the development of new text-to-shape coherence metrics.

2.2 Related Work

Text-to-shape coherence metrics. A large number of studies have addressed the task of text-conditioned 3D generation. Most of them have proposed quantitative metrics, which, however, are only able to measure the generative capability of the methods, i.e., the realism of the generated shapes. To assess

fidelity to the input prompt [76, 107, 20] compute several distance measures between a generated shape and the one associated with the input prompt in the ground truth set, such as Intersection-over-Union (IoU), Earth’s Mover Distance (EMD), Chamfer Distance (CD) and Mean Squared Error (MSE). The main limitation of such a strategy is that, in generative tasks, a ground truth shape is just one possible correct outcome of the generation process and relatively higher or lower distances from it do not measure the quality of the output nor, in the case of text-conditioned generation, its coherence with the given text. The most convincing metrics for text-to-shape coherence are based on CLIP embeddings [99]. One such metric is CLIP R-precision [94], used in several studies [92, 76, 95, 52]. It is defined as the accuracy with which CLIP retrieves the correct caption among a set of distractors given a rendering of the generated shape. Yet, such quantitative assessment is affected by several limitations: first of all, it is based on renderings, therefore it is influenced by a number of parameters, like virtual camera placement, virtual illumination, and number of rendered views; secondly, the selection of the text distractors is arbitrary; finally, as it relies purely on rendered views, whose quality is not homogeneous among 3D data representations, a comparison between 3D shapes from different representations (e.g. point clouds and meshes) can be unfair. CLIP-similarity [36], instead, computes the cosine-similarity between CLIP image features extracted from several shape renderings and CLIP text features. Due to the reliance on renderings, this metric shares the same limitations of CLIP R-precision.

ShapeCrafter [36] has been the first work to use ShapeGlot [2] as a text-shape fidelity metric. In its original formulation, given a set of shapes, this metric can discriminate which one is the most coherent with respect to an input text description by extracting 3D features, 2D features from renderings, and text features. Despite its usefulness, there are some clear disadvantages associated with this method. Firstly, similar to CLIP-based metrics, its effectiveness relies on the parameter settings used to render the views of the shape. Secondly, the text representation employs an LSTM architecture, which has been acknowledged to have difficulties in representing long text descriptions [14].

To address all the above limitations, we propose *CrossCoherence*, a metric for text-to-shape coherence that does not depend on rendering parameters, operates directly on colored point clouds to take both geometry and appearance into account, leverages LLMs to extract better text features, and exploits cross-attention to assess the coherence of words with shape parts.

Datasets. Several text-driven 3D generation models require supervision in the form of text-shape pairs. Unlike in the 2D case, where web scraping can be used to obtain large amounts of image-text pairs [22, 121, 116, 108], obtaining such labeled data is extremely difficult, due to 3D shapes not being common on the Internet and the complexity and cost of creating text descriptions for large 3D datasets like ShapeNet [17]. Text2Shape [20] has been the first dataset with shape-text pairs to be introduced. It contains text prompts for two Shapenet classes: chairs and tables. Overall, Text2Shape provides 75K shape-text pairs, with multiple text descriptions for each 3D object. However, the quality of these descriptions is highly variable: the prompts are either too generic (*“A chair with four legs”*) or contain irrelevant details (*“Table with high-quality wood, you can eat and enjoy with your family”*) or both. Sometimes the text does not correctly describe the geometry and/or appearance of the objects (*“A table with three legs”* but the table has actually four legs). In addition, many sentences contain grammatical and syntactical errors. These nuances make the process of learning the complex relationship between 3D data and text even more difficult and limit the validity of comparisons carried out on this dataset. Text2Shape++ [36] is a dataset built upon Text2Shape, including 369K shape-text pairs. This dataset has been built specifically for the task of recursive shape generation; indeed, the original text descriptions from Text2Shape have been split into multiple smaller sentences with incremental amount of information (e.g. *“a chair”*, *“a chair with four legs”*, *“a chair with four legs and no armrests”*). Since Text2Shape++ is built starting from the same descriptions of Text2Shape, its text prompts suffer from the same limitations.

In response to these challenges, we propose an automatically improved version of Text2Shape, referred to as GPT2Shape, which contains new text descriptions for every object of Text2Shape. These novel sentences can describe more accurately the geometry, appearance and texture of all the objects in the dataset, as proved by our human evaluation study and exemplified in Figure 2.2.

2.3 GPT2Shape

As can be seen from this Figure, the quality of the original descriptions is highly variable and some of them are quite poor: for instance, the sentence *“grey desk”* is too generic and not able to drive any generative model towards the intended 3D shape. Improving such a dataset through a new human-based

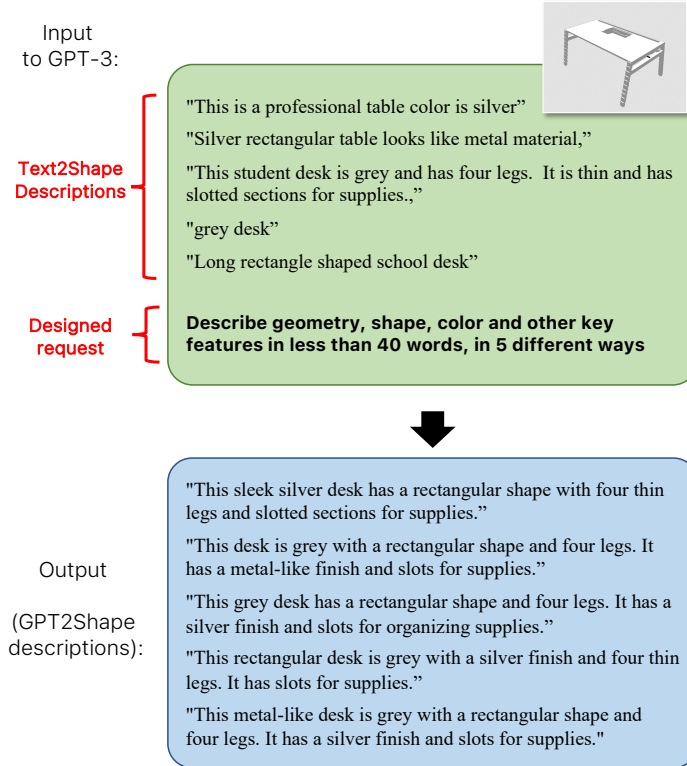


FIGURE 2.2: An example of the text prompts rephrasing process. The green box contains Text2Shape descriptions for a shape together with the designed request; their combination represents the input prompt to GPT-3. The blue box is the output of GPT-3, providing the textual descriptions of GPT2Shape.

annotation on the Amazon Mechanical Turk crowd-sourcing platform [27], utilized by Text2Shape, would be extremely time-consuming and expensive.

At the same time, enough details about a shape are usually present in Text2Shape if information from multiple prompts are merged, as shown again in Figure 2.2. On the basis of this observation, we decided to explore if an automatic improvement of the existing dataset was feasible, by exploiting LLMs. In particular, the generation of the new text prompts has been achieved through the OpenAI GPT-3 model [15] *davinci*. GPT-3 has been used in its text completion scenario in this manner: for each shape of the dataset, all its corresponding text prompts from Text2Shape have been given as input to the language model, along with the request to generate better text descriptions. The request itself was the result of a careful prompt engineering process, in order to obtain the best possible result. Figure 2.2 illustrates an example of this process, showing the initial text descriptions, the request to GPT-3, and the resulting sentences. As it can be noticed, the output textual descriptions contain all the relevant geometric and appearance information regarding the object. In addition, any contradictions contained in the source sentences have

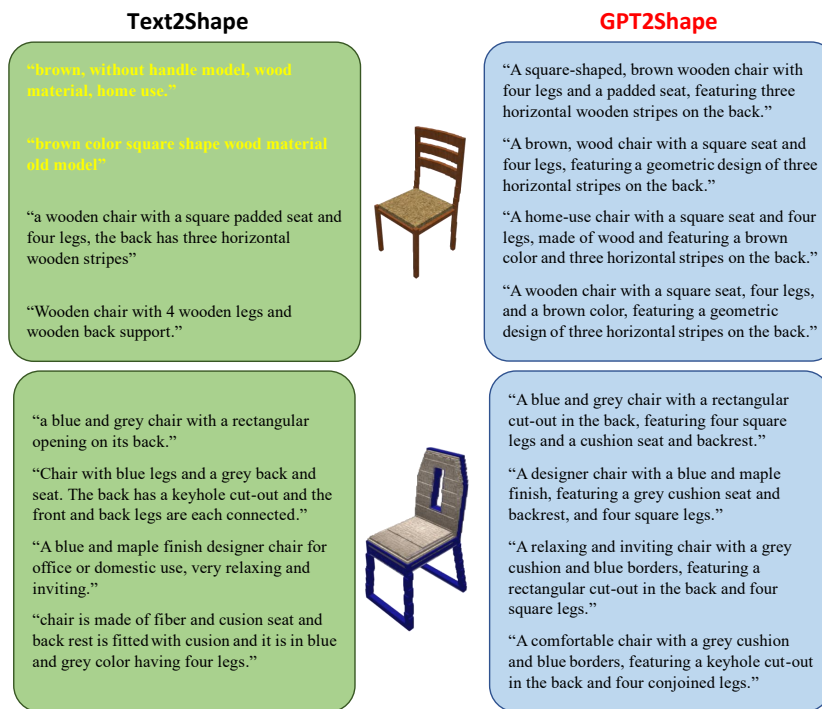


FIGURE 2.3: Example of Text2Shape and GPT2Shape textual annotations. The green box contains the original Text2Shape descriptions for the 3D shape in the middle. The blue box shows the rephrased text prompts of GPT2Shape. In yellow, we highlight the ambiguous or wrong descriptions of Text2Shape.

been automatically filtered by the LLM during the text generation process. This text rephrasing process was carried out on the whole train-val-test splits of Text2Shape, so as to keep the same amount of data samples as the original dataset: 15032 shapes, with a total of 75358 shape-text pairs. In Figure 2.3, an example of the textual annotations from Text2Shape and GPT2Shape is provided.

2.4 GPT2Shape vs Text2Shape: a user study

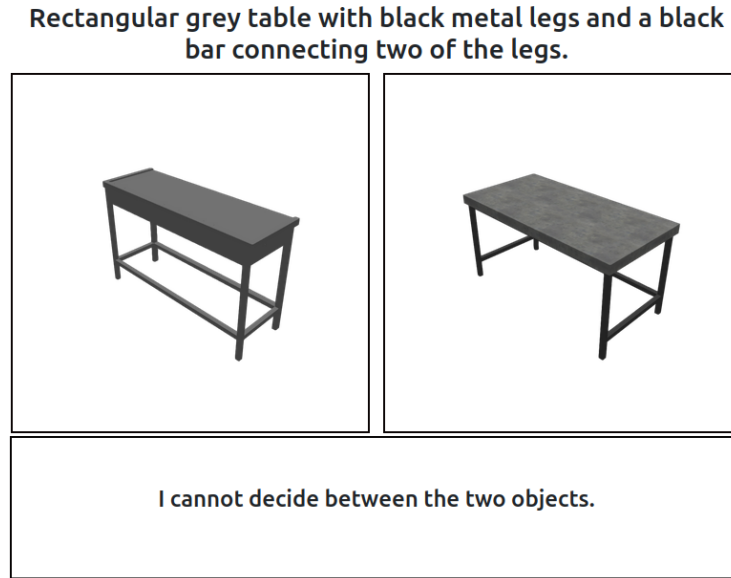


FIGURE 2.4: Layout of the user study

Once the rephrased sentences had been obtained via GPT-3, a user study aimed at comparing the informativeness of the old (Text2Shape) and new (GPT2Shape) text prompts was carried out. The layout of this user study is shown in Figure 2.4: each user was shown a pair of views from two objects together with a text prompt. The prompt comes from the test set of either Text2Shape or GPT2Shape and describes one of the two objects (i.e., the reference object, unknown to the user) while the other one acts as a distractor. The left/right position of the reference object was randomized. The task consisted in clicking on the object which, according to the user, would be described better by the given text. In case of uncertainty, due to the prompt describing equivalently well both objects or not sufficiently well any of the two, the user could click on a third option, located under the images. This enabled us to compare the two datasets based on users' choices: the higher the number of clicks on the -unknown- reference object, the higher the informativeness of the associated text prompt.

To avoid presenting the user with a task far too simple, as it might have been the case had distractors been chosen randomly, the *reference-distractor-text* triplets were built as follows. For every reference object, we have defined two *hard* distractors and one *easy* distractor. All distractors for a shape have been selected within its class (e.g., chairs or tables) and defined based on the Euclidean distance in the latent space of a point cloud autoencoder. The architecture of this network is reported in Figure 2.5. This autoencoder is

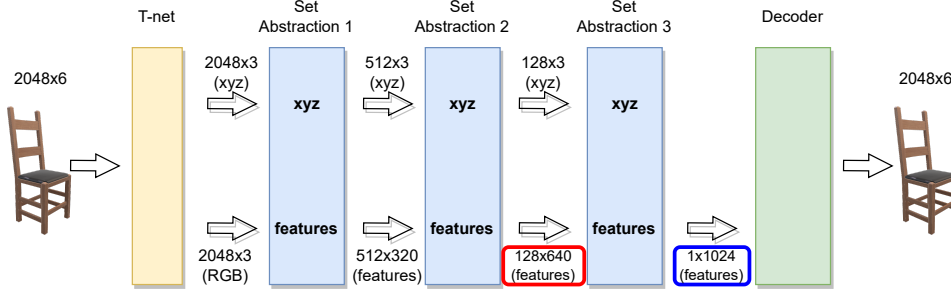


FIGURE 2.5: Architecture of the PointNet++ autoencoder. In **red** the features used by *CrossCoherence*, and in **blue** the embeddings that have been used to compute the *easy* and *hard* distractors for each reference shape.

composed of a transformation network T-Net [96], a PointNet++ [97] encoder with three set abstraction layers, and a simple convolutional decoder, made of three stacked 1×1 convolutions with ReLUs in between. Different from a standard PointNet++ encoder, the input of this network is an RGB point cloud featuring spatial coordinates (x, y, z) and colors (R, G, B) , which are used as input features for every point. The model has been trained on the task of shape reconstruction on the chair and table categories of ShapeNet [17]. To build the *easy* and *hard* shape distractors, we extracted the shape embeddings from the third set-abstraction layer, which is a global feature vector of dimension 1024.

Leveraging the two *hard* distractors and one *easy* distractor for each reference object, we built three pairs of 3D shapes. When running the user study, one pair of shapes was randomly sampled from this set, whereas the text describing the reference shape was taken from Text2Shape or GPT2Shape with 50% probability. In the publication [5], we provide more details and examples on easy-hard shape distractors and triplets from the user study.

Overall, 175 users took part in the study, contributing with 3642 data samples (i.e., clicks). In order to evaluate Text2Shape and GPT2Shape, we kept track of the answers provided by the users when presented with text prompts from both datasets. Table 2.1 summarizes the results by reporting the percentages of times users selected the reference object, the distractor, or felt they could not decide when presented with a text from Text2Shape or GPT2Shape. When the text came from GPT2Shape, about 75% of the answers were correct, i.e. the human selected the reference shape as the most coherent with the provided text description. On the other hand, Text2Shape allowed the users to answer correctly only 65% of the time. Moreover, the percentage of undecidable sentences is higher for Text2Shape (about 25%) compared to GPT2Shape (about 16%).

Such experimental findings validate our claim that large language models

Dataset	reference ($\uparrow$)	cannot decide ($\downarrow$)	distractor ($\downarrow$)
Text2Shape	65.65%	24.95%	9.40%
GPT2Shape	75.76%	16.21%	8.02%

TABLE 2.1: Summary of user study results

can be used to automatically filter and combine information from multiple incomplete textual descriptions. They also validate that textual descriptions in GPT2Shape are more informative than the original prompts from Text2Shape.

2.5 Human-validated Shape-Text dataset

The results of the user study were also used to aggregate a refined test set of shape-text pairs with descriptive texts from the test sets of Text2Shape and GPT2Shape, which we dub the **Human-validated Shape-Text (HST)** dataset. When building this subset, we only selected the sentences for which all users made the same choice between the two objects, i.e., where there was unambiguous consensus among participants that the description described well only one of the two shapes. HST, which contains 2153 text-shape pairs, provides a human-validated benchmark to assess the quality of text-to-shape generative models. It can be used as a set of descriptive text prompts to evaluate such models with text-to-shape coherence metrics. To provide a comprehensive dataset, for every pair in the HST dataset, we also included the other object that was shown to the human evaluators, along with the indication of the correct association for the text. This extended version of HST is useful for quantitatively validating metrics for text-to-shape coherence, as we do in the next section.

2.6 CrossCoherence

In this Section, we describe the details of our proposed metric for text-to-shape coherence, *CrossCoherence*. Recent works on text-conditioned generative models have demonstrated that frozen Large Language Models (LLMs) such as BERT [33], GPT [15], and T5 [100], despite being trained solely on text, serve as highly effective text encoders for multimodal tasks. Notably, Imagen [106] showed that these LLMs can outperform encoders specifically trained on paired image-text data, such as CLIP [99]. Furthermore, the success of such architectures is often attributed to sophisticated text-conditioning schemes based on the cross-attention mechanism. Motivated by these observations, we

explore the integration of a frozen LLM with bilateral cross-attention layers to define a robust 3D fidelity metric.

2.6.1 Architecture and Training Objective

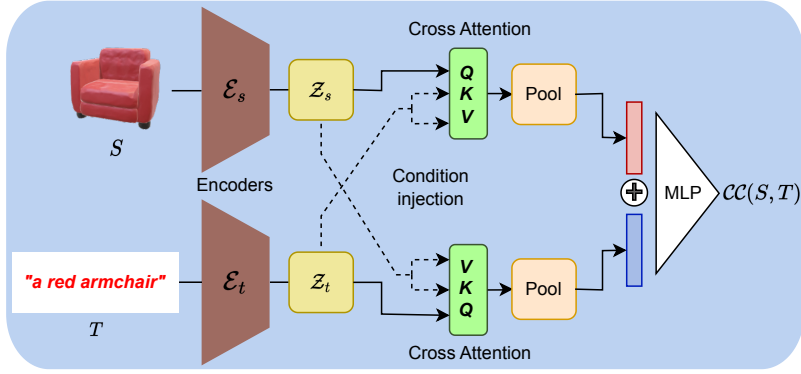


FIGURE 2.6: Architecture of *CrossCoherence*

The proposed architecture of *CrossCoherence* is illustrated in Figure 2.6. The model takes as input a text description and a pair of N colored point clouds, characterized by coordinates (x, y, z) and color values (R, G, B) . The architecture is designed to extract features from both modalities and reason over their correspondence to predict which point cloud best aligns with the input text.

Feature Extraction. We employ the encoder portion of a PointNet++ network [97] as our Point Cloud Encoder, pre-trained for shape reconstruction on the ShapeNet chairs and tables categories. While the distractor generation process described in Section 2.4 utilizes global embeddings from the third set abstraction layer, *CrossCoherence* leverages the second set abstraction layer to preserve local features on the downsampled point cloud. For the Language Encoder, we utilize the frozen encoder from T5 [100] to extract high-dimensional textual embeddings.

Bilateral Cross-Attention. To evaluate coherence, the model employs two *bilateral* cross-attention layers for every point cloud. These layers are designed to capture the mutual relationship between words and 3D details: the first layer uses shape features as queries and text embeddings as keys and values, while the second layer reverses this roles. This mechanism allows the model to perform fine-grained reasoning on how specific words relate to local geometric features. The resulting embeddings are average-pooled, concatenated, and fed into an MLP to produce a raw coherence score s_i for each point cloud.

Loss Function. During training, the scores s_i for each point cloud $i \in \{1, \dots, N\}$ are normalized using a softmax function to obtain predicted probabilities $\hat{s}_i$:

$$\hat{s}_i = \frac{\exp(s_i)}{\sum_{j=1}^N \exp(s_j)} \quad (2.1)$$

The model is supervised using a categorical cross-entropy loss:

$$\mathcal{L} = - \sum_{i=1}^N y_i \log(\hat{s}_i) \quad (2.2)$$

where $y_i \in \{0, 1\}$ denotes the ground-truth correspondence between the input text and the i -th point cloud.

Discussion on the supervision. While contrastive objectives, such as the InfoNCE loss used in CLIP [99], are the standard for multimodal alignment, we opted for cross-entropy supervision for two primary reasons. First, **data efficiency**: contrastive learning typically requires massive datasets and large batch sizes to provide sufficient "hard negatives" to prevent representation collapse. Given our constrained 3D-text dataset, cross-entropy provides a more stable supervisory signal. Secondly, **fine-grained reasoning**: contrastive losses excel at global "coarse" alignment. However, 3D fidelity evaluation requires fine-grained, local interaction between modalities. By treating the task as a discriminative selection problem, our model is forced to reason on the specific structural differences between candidates, a task-specific focus that directly aligns with our inference setting.

2.6.2 Training

During training, the network is fed with $(shapes, text)$ pairs and target vectors indicating the ground truth shape corresponding to the textual description. Based on the outcome of the user study, we train *CrossCoherence* on the training set of GPT2Shape. Before training, following the same strategy applied for the human evaluation survey, described in Section 2.4, for every shape in the training set of GPT2Shape, we extract one easy and two hard distractors. This *easy-hard* training strategy was found to be more effective than constructing the pairs randomly due to the network learning to deal with cases where the input point clouds are very similar to each other, which in turn is conducive to better generalization to unseen examples.

Method	Accuracy on chairs ($\uparrow$)	Accuracy on full HST ($\uparrow$)
ShapeGlot	70.21%	-
CLIP-Similarity	77.79%	77.06%
<i>CrossCoherence</i>	81.04%	80.45%

TABLE 2.2: Comparison between ShapeGlot, *CrossCoherence* and CLIP-Similarity. The first column reports accuracy for methods trained and tested only on chairs, while the second column is for methods trained and tested on the full HST.

Test set	ShapeGlot	<i>CrossCoherence</i>	CLIP R-precision
HST chairs	5.36%	17.26%	9.87%
full HST	-	16.85%	9.38%

TABLE 2.3: Comparison between ShapeGlot, *CrossCoherence* and CLIP R-precision based on the R-precision protocol. The first row reports the results for methods trained and tested only on chairs, while the second row is for methods trained and tested on full HST.

2.7 Experimental Results

We performed several experiments to assess the effectiveness of *CrossCoherence*.

2.7.1 Comparison with existing metrics

In this section, we evaluate the accuracy in assessing text-to-shape coherence for our proposed *CrossCoherence* metric, as well as for Shapeglot [2], CLIP-Similarity [36, 99], and CLIP R-precision [94, 99]. The accuracy computation relies on the triplets of HST (Section 2.5) and consists of two different evaluation protocols. When comparing with CLIP-Similarity, we report how often each metric yields a higher score to the ground truth shape than to the other one in the triplet for the given text description. In contrast, when dealing with CLIP R-precision, we adopt the protocol introduced in [50] and used by [92, 76, 95]: given a text prompt and the corresponding ground truth 3D shape from HST, we construct a set of 153 text descriptions. This set comprises the ground truth prompt and 152 other texts randomly sampled from a designated test set, in our case, HST. Every metric will predict which text prompt within the set exhibits the highest coherence to the given 3D shape. By comparing the predicted prompt to the ground truth prompt, we can assess the performance of the evaluation metric. To evaluate both CLIP-Similarity and CLIP R-precision, we conduct experiments using various CLIP base models and different numbers of rendered views. When dealing with more than one rendered view, we used the maximum similarity across the views, as in [36].

The configuration that achieves the best performance, which we consider here, relies on ViT-L/14 CLIP Encoder and 20 renderings. Yet, we deem it worth highlighting that the accuracy of CLIP-based metrics varies significantly depending on the chosen configuration (CLIP base model, 3D data representation, rendering parameters), as shown in additional experiments reported in the publication [5]. As for ShapeGlot [2], we train the network on the same dataset as *CrossCoherence*, i.e., easy and hard samples from GPT2Shape, as described in Section 2.6. However, while the training of *CrossCoherence* relies only on point clouds and text prompts, ShapeGlot also consumes rendered views of objects’ meshes and requires a pre-trained VGG encoder to extract features from these renderings. Unfortunately, the authors did not release the weights of the VGG model and made available only pre-computed features for the *chair* class of Text2Shape. Hence, we can assess the accuracy of ShapeGlot only on the chair subset of HST.

The outcome of our evaluation is shown in Tables 2.2 and 2.3. These results indicate that *CrossCoherence* is the most effective metric. We can observe how the performance of ShapeGlot is the least satisfactory. Moreover, the gap in accuracy between this metric and *CrossCoherence* suggests that modern text-analysis tools like LLMs and cross-attention layers are key to creating a more effective metric. Indeed, the main differences between *CrossCoherence* and ShapeGlot include the use of LLMs instead of LSTMs as well as the incorporation of the cross-attention mechanism. Finally, in both settings, CLIP-based metrics turn out to be the strongest competitors; however, *CrossCoherence* outperforms CLIP-Similarity by more than 3% and is more accurate than CLIP in the challenging R-precision setting with a gap around 7%, being the most widely used protocol for evaluating text-to-shape coherence.

Qualitative examples of the choices made by CLIP-Similarity and *CrossCoherence* are provided in Figure 2.7. The errors made by CLIP-Similarity seem to be caused by the use of global embeddings to compare shapes and text, which prevents the metric from evaluating fine details when judging the shape-text agreement. For instance, CLIP-Similarity does not take into account the shape of the table legs in the first row, the presence of the chair cushion in the second row and the geometry of the glass table in the last example. As shown in Figure 2.8, CLIP R-precision suffers from the same weakness as CLIP-Similarity in understanding fine-grained details of visual input and text. For example, CLIP R-precision provides high scores for text prompts that contain multiple wrong references to the geometry and appearance of the input shape, i.e. “Rectangular wooden table” in the first row, “Armless rocking

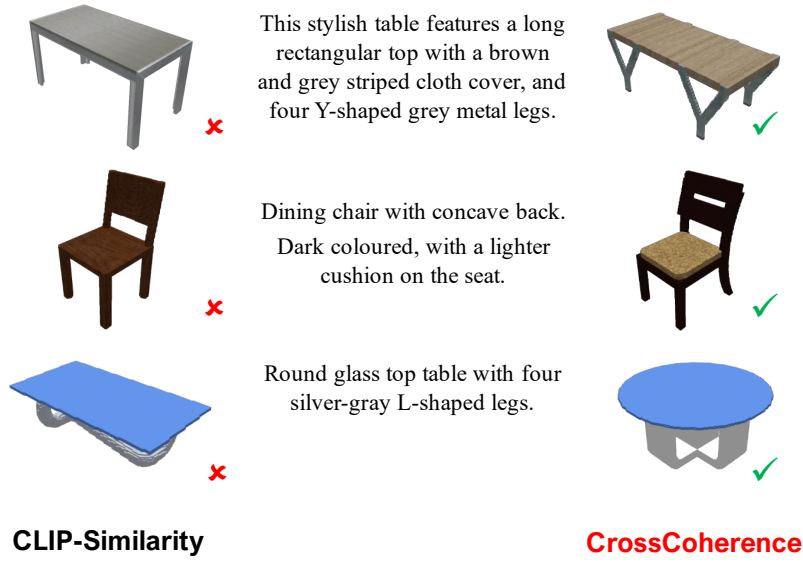


FIGURE 2.7: Qualitative results of CLIP-Similarity and *CrossCoherence* on HST dataset. The green check indicates the shape associated with the prompt, while the red cross identifies the distractor. For all these triplets, CLIP-Similarity prefers the left shape while *CrossCoherence* the right one.

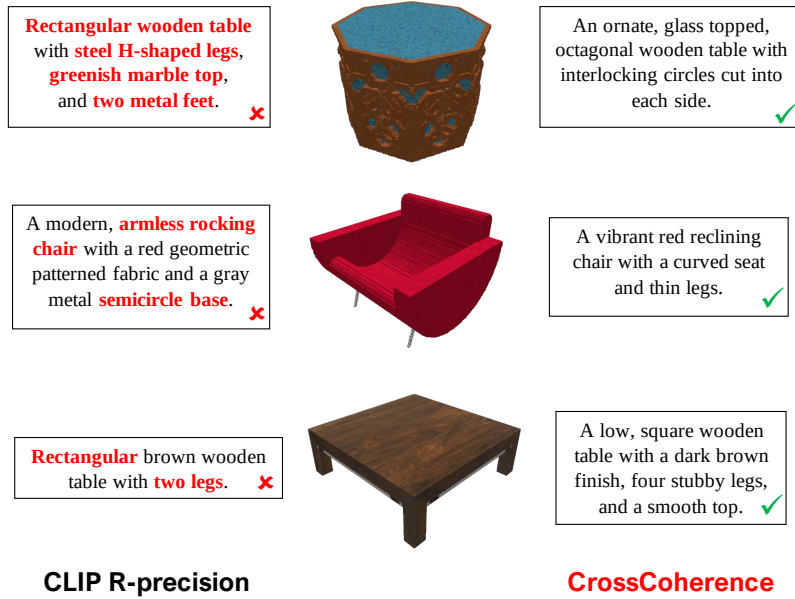


FIGURE 2.8: Qualitative results of CLIP R-precision and *CrossCoherence* on HST dataset. The green check indicates the ground truth text. For all these triplets, CLIP R-precision retrieves the left text while *CrossCoherence* the right one.

chair” in the second example, “*two legs*” in the last row. This weakness has been highlighted in [124] and [138], which discuss how CLIP models, probably due to their contrastive pretraining strategy, have difficulty in processing fine-grained descriptions of visual content. Indeed, the performance of such models on the Attribution, Relation, and Order (ARO) benchmark [138] show

Method	CrossCoherence	CLIP-Similarity	CLIP-Similarity
Point-E [92]	27.33%	24.36%	27.04%
Shap-E [52]	38.02%	65.47% (STF)	57.18% (NeRF)
Liu et al. [76]	34.65%	10.18%	15.78%

TABLE 2.4: Comparison of *CrossCoherence* and CLIP R-precision across generative models. Each entry in a column reports how often a metric considers a prompt more coherent with the shape generated by one method versus the others.

that CLIP exhibits poor relational understanding and tends to behave like bag-of-words models. The use of cross-attention between local embeddings in *CrossCoherence*, on the other hand, enables to capture and link together important shape and text details, leading to the selection of the reference shape or text, as shown in Figures 2.7 and 2.8, respectively.

2.7.2 Text-to-shape coherence of generative methods

A text-to-shape coherence metric can be used to evaluate text-conditioned 3D shape generative models. Here we address this setting and compare *CrossCoherence*, CLIP-Similarity [36] and CLIP R-precision [94] in their ability to judge upon the coherence to the input text of the shapes generated by *Point-E* [92], *Shap-E* [52] and the model proposed by Liu et al. [76]. These models employ distinct data representations, with Point-E being a method for text-driven point cloud generation, Shap-E using both NeRF (neural radiance field) and STF (signed textured field), Liu et al. [76] employing occupancy fields. The goal of this experiment is to show an example of the use of our metric and to demonstrate its improved ability to capture shape-text coherence, compared to CLIP-based metrics, in shapes generated by state-of-the-art models.

We used the pretrained weights provided by the authors for Point-E and Shap-E, while we trained from scratch Liu et al. [76] on GPT2Shape. We then used them to generate one shape for each prompt in the HST dataset and used *CrossCoherence*, CLIP-Similarity and CLIP R-precision to decide which shape was more coherent with the input prompt. We adopted the same experimental protocols as in Section 2.7.1, the only difference being that here the 3D shapes compared to the text prompts are not the ground truths from HST, but they have been generated by either Shap-E or Point-E or Liu et al.

Results are reported in Tables 2.4 and 2.5.

Since Shap-E represents the generated shape as an STF and a NeRF, for CLIP-Similarity and CLIP R-precision, we report the results computed on renderings from both representations. The differences in performance for

Method	CrossCoherence	CLIP R-precision	
Point-E [92]	4.92%	5.95%	
Shap-E [52]	7.09%	20.41% (STF)	15.78% (NeRF)
Liu et al. [76]	5.09%	5.83%	

TABLE 2.5: Comparison of CrossCoherence and CLIP R-precision across generative models.

Shap-E suggest again that CLIP-based metrics are significantly affected by the rendering process. In these experiments, we notice that all metrics agree on Shap-E being the method that generates the 3D shapes closest to the input text. However, the ranking of the other two methods in relation to Shap-E is very different. In fact, *CrossCoherence* ranks the method of Liu et al. [76] second, with a modest gap of 4% in the first protocol and 2% in the second. In contrast, CLIP-similarity and CLIP R-precision rank Point-E in second place with a very large gap in performance with respect to Shap-E (around 30% in the first protocol, and from 10% to 15% in the second one).

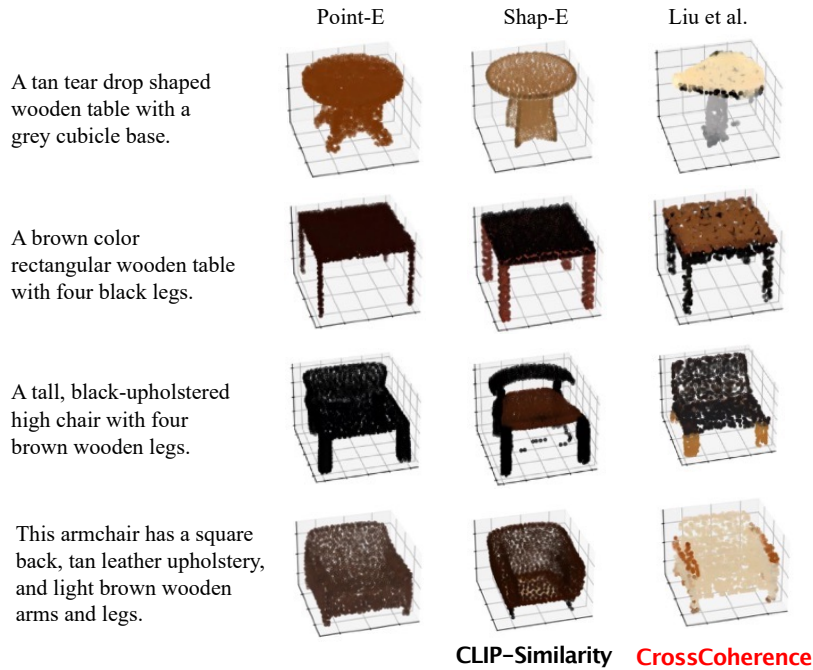


FIGURE 2.9: Examples of shapes generated by Point-E (left), Shap-E (center) and Liu et al. (right) for the given text prompt. As highlighted, for all these triplets, CLIP-Similarity prefers the shape in the middle while *CrossCoherence* the one on the right.

By inspecting some triplets on which *CrossCoherence* and CLIP-Similarity disagree, shown in Figure 2.9, we can notice how CLIP-similarity seems unable to recognize certain geometric and color details, present in the shapes generated by Liu et al. [76], which are critical for determining the shape

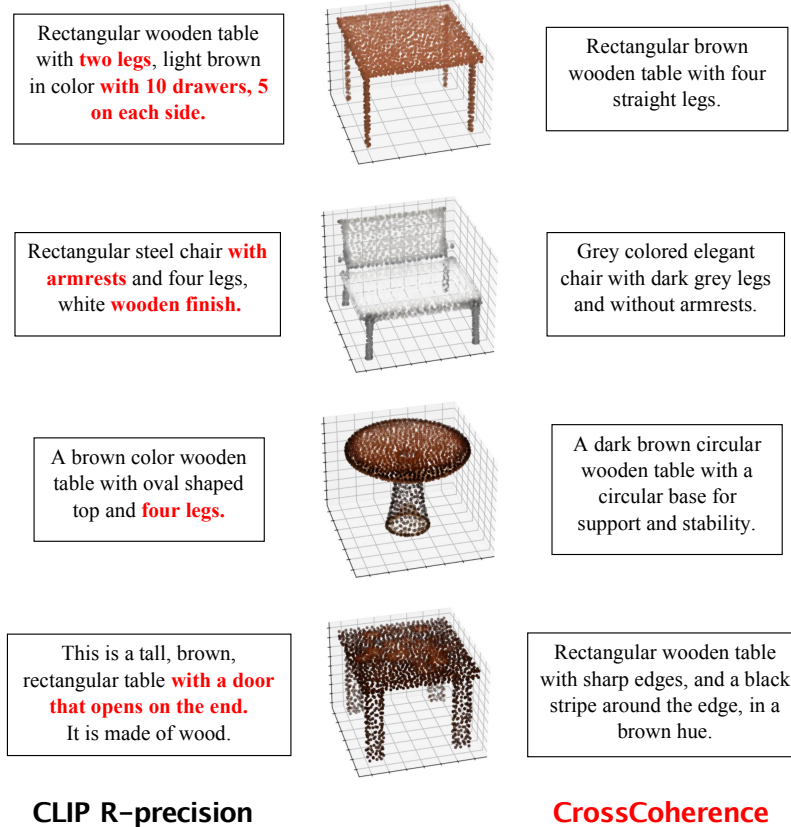


FIGURE 2.10: Examples of shapes generated by Shap-E with the corresponding text retrieved by CLIP R-precision, on the right, and *CrossCoherence*, on the left.

most coherent to the given text prompt. In the first row of Figure 2.9 CLIP-Similarity prefers the table in the middle although it lacks the “grey cubicle base” and the “tear drop shape” described in the text. In the second row, CLIP-Similarity is not able to localize the parts with the colors specified in the text. As discussed in Section 2.7.1, such weakness may be caused by the difficulty of CLIP in processing fine-grained text descriptions and understanding the spatial relations among the objects’ parts.

In Figure 2.10 we report some qualitative examples to motivate the large gap between *CrossCoherence* and CLIP R-Precision. Here, it is possible to observe that even if in some cases CLIP R-precision retrieves the ground truth text as the most coherent with the shape from Shap-E, these descriptions contain details that are not present in the generated shape (e.g., “two legs” in the first row, “with armrests” in the second row, “four legs” in the third row and “with a door that opens on the end” in the last row). This behavior arises because Shap-E fails to generate all the specified details in the ground truth text and CLIP R-precision cannot accurately recognize the absence of these elements in the generated shape. On the contrary, *CrossCoherence* can retrieve

more comprehensive and accurate text prompts which are more coherent with the generated shapes.

2.8 Ablation experiments on the attention layers of *CrossCoherence*

In Table 2.6, we show the performance achieved by *CrossCoherence* using different attention schemes. In particular, we evaluate the following architectures:

1. *Without Attention*: in this architecture, the shape and text embeddings directly undergo an average pooling layer, before being concatenated. No attention layer is present.
2. *Bilateral Cross-Attention*: this is the original scheme for *CrossCoherence*, featuring two cross-attention layers where in the first one the queries of the attention maps are computed on the shape features, whereas the keys and values are obtained from the text embeddings, while in the second layer this computation is reversed. Finally, the resulting embeddings from both the layers are concatenated, after undergoing an average pooling layer.
3. *Self-Attention*: we replace the cross-attention layers with self-attention layers and perform pooling and concatenation of their results.
4. *Bilateral Self-Attention and Cross-Attention*: before performing cross-attention, we feed the queries of each layer into a self-attention layer (one for the shape features and one for the text features).
5. *Bilateral Cross-Attention and Feed-Forward*: after performing cross-attention, we process the queries of each layer with a feed-forward layer (one for the shape features and one for the text features).

The Table points out clearly that the proposed Bilateral Cross-Attention (first row) is more effective than all the other formulations considered in the experiments. This suggests that the Cross-Attention mechanism effectively captures meaningful relationships between shape and text features, leading to improved performance. The models utilizing Self-Attention alone and the combination of Self-Attention and Cross-Attention achieved very similar results, just below the best architecture. Self-Attention allows the model to attend to different parts of the input sequence, but it seems to be less

effective in capturing complex dependencies compared to Cross-Attention. However, when equipped with a cross-attention layer, the performance had very slight variations on both the subset and the full HST, suggesting that adding Cross-Attention in this configuration did not provide any substantial benefit. Interestingly, the combination of Cross-Attention and Feed-Forward yielded the second-highest accuracy on the chairs of HST, 80.28%, but obtained performance comparable to the other schemes on the full HST. Finally, the architecture without any attention layer achieved the lowest accuracy on both datasets.

These findings highlight the importance of carefully selecting attention mechanisms in the design of models, with Cross-Attention being a particularly valuable choice for this specific task.

Attention layers	Accuracy on chairs (↑)	Accuracy on full HST (↑)
No Attention	75.95%	74.54%
Cross-Attn	81.04%	80.45%
Self-Attn	78.98%	78.08%
Self-Attn and Cross-Attn	78.76%	78.40%
Cross-Attn and Feed-Forward	80.28%	77.66%

TABLE 2.6: Comparison on the HST test set between *CrossCoherence* using different attention schemes. The second column specifies the kind of attention used in the model. The third column reports accuracy for models trained and tested only on chairs, while the fourth column shows the results for models trained and tested on both chairs and tables.

2.9 Sensitivity of CLIP-based metrics to different 3D representations

We have carried out a sensitivity study on the 3D representation adopted to compute CLIP-Similarity and CLIP R-precision, by comparing the results achieved on HST with renderings obtained from point clouds and meshes. Numbers are reported in Tables 2.7 and 2.8.

These results highlight the strong dependence of CLIP-based metrics on the used 3D data representation. In particular, Tables 2.7 and 2.8 suggest that point clouds should be transformed into meshes to enable effective evaluation of text-to-shape coherence. Conversely, *CrossCoherence* is designed to work natively on point clouds and can judge effectively upon this kind of data without requiring conversion into meshes. This behavior of CLIP-based metrics can make them less generalizable across different types of 3D data and potentially biased towards certain types of representations. This can

lead to inaccurate similarity scores and limit the usefulness of such metrics in applications that require robust similarity measures.

Data representation	Accuracy on chairs ($\uparrow$)	Accuracy on full HST ($\uparrow$)
Mesh	77.79%	77.06%
Point cloud	72.26%	70.78%

TABLE 2.7: Sensitivity study of CLIP-similarity to different 3D representations.

Test set	Data representation	CLIP R-precision
HST chairs	Mesh	9.38%
HST chairs	Point cloud	8.95%
full HST	Mesh	9.87%
full HST	Point cloud	7.83%

TABLE 2.8: Sensitivity study of CLIP R-precision to different 3D representations.

2.10 Sensitivity of CLIP-based metrics to CLIP base model and number of renderings.

Given the same 3D representation, the performance of CLIP-based metrics as text-shape coherence metrics is strongly influenced by two factors: first, the CLIP base model used to compute the image and text embeddings; second, the number of rendered views used to compute the metric. Table 2.9 summarizes the results of CLIP-Similarity obtained using two different CLIP models and considering 1 or 10 or 20 rendered views. We report the two CLIP models that provided the best performance, with *ViT-L 14* performing slightly better than *ViT-B 32*. The rendered views are obtained from the mesh of every 3D shape. The gap in performance between the best and worst configuration is larger than 6% for CLIP-Similarity..

CLIP model	Renderings	Accuracy on chairs ($\uparrow$)	Accuracy on full HST ($\uparrow$)
ViT-B 32	1	73.46%	70.51%
ViT-B 32	10	76.49%	75.43%
ViT-B 32	20	77.14%	75.89%
ViT-L 14	1	72.91%	70.32%
ViT-L 14	10	77.03%	76.73%
ViT-L 14	20	77.79%	77.06%

TABLE 2.9: Performance of CLIP-Similarity on HST dataset using different CLIP base models and a different number of rendered views.

Since both CLIP R-precision and CLIP Similarity rely on the same vision-language model (e.g., CLIP) the sensitivity issues we discussed earlier regarding CLIP Similarity also apply to CLIP R-precision. In other words, both metrics are sensitive to the data representation, the CLIP-base model and the number of rendered views.

2.11 Conclusion

In this Chapter, we have addressed two of the most significant bottlenecks in the text-to-3D field: the scarcity of high-quality, fine-grained datasets and the lack of reliable metrics for evaluating text-shape coherence. By leveraging Large Language Models to curate GPT2Shape and introducing the *CrossCoherence* metric, we have demonstrated that effective 3D-language alignment requires moving beyond global embeddings (like CLIP) toward a more localized, cross-attention-based understanding of how specific words correspond to geometric and appearance-based details in a point cloud.

The development of *CrossCoherence* provided a crucial insight: 3D-language understanding is highly sensitive to the underlying representation of the 3D data. While point clouds offer a direct way to model geometry and color, they remain a discrete and often sparse representation. The challenges encountered in Chapter 1, specifically the difficulty of capturing fine-grained coherence from noisy or resolution-dependent data, motivated a shift in perspective. If a model is to truly "understand" an object, perhaps it should not rely on sampled points or 2D renderings, but rather on the continuous, high-fidelity information stored within the neural representation itself.

This realization serves as the catalyst for the following chapters. Having established a framework for evaluating 3D-text coherence, we now transition from evaluating alignment in discrete point clouds to architecting systems that can inherently "reason" about 3D objects. In Chapter 2, we move beyond the limitations of point clouds and explore Neural Radiance Fields (NeRFs) as a primary input for Multimodal Large Language Models. We will investigate whether the weights of these neural representations, which compactly encode the full geometric and appearance-based essence of a scene, can be directly ingested by an LLM to create a more robust and versatile 3D assistant.

Chapter 3

The Rise of NeRF-language Models

This Chapter refers to the works "*LLaNA: Large Language and NeRF Assistant*" [4], published at the peer-reviewed Conference on Neural Information Processing Systems (NeurIPS) 2024, and "*Scaling LLaNA: Advancing NeRF-Language Understanding Through Large-Scale Training*" [6], currently under review for the journal "IEEE Transactions on Pattern Analysis and Machine Intelligence".

3.1 Introduction

The field of Natural Language Processing has been profoundly transformed by Large Language Models (LLMs) [33, 100, 1, 122], due to their text comprehension and generation capabilities. These results have fostered the development of Multimodal LLMs (MLLMs) [74, 142, 44], which can process various modalities such as images, videos, and audio, to generate text describing and reasoning about the content of such modalities. Recently, MLLMs have also been extended to 3D data [133, 98, 43], primarily represented as colored point clouds, yielding remarkable results even in this scenario.

Another approach for representing objects and scenes has emerged alongside traditional images and 3D data: Neural Radiance Fields (NeRFs) [88]. NeRFs are coordinate-based neural networks, typically Multi-Layer Perceptrons (MLPs), designed to capture both the geometry and the photorealistic appearance of an object. By learning a continuous radiance field across 3D space, NeRFs can be used to generate realistic images from any viewpoint or reconstruct the 3D surface of an object, by querying the trained model. Using NeRFs to represent 3D data offers distinct advantages over conventional approaches like multi-view images or point clouds. The continuous nature of NeRFs allows generating unlimited photorealistic images at any desired resolution while only requiring the storage of MLP weights rather than a large collection of images. Moreover, unlike point clouds that lack

information about texture, lighting, and material properties, NeRFs explicitly model view-dependent appearance, enabling realistic rendering effects such as specular highlights and transparency. They also provide a unified representation that seamlessly combines geometry and appearance, avoiding the need for separate texture maps or mesh-based pipelines. In addition, the neural representation is compact and compressible, often requiring less memory than dense voxel grids or high-resolution images, while still supporting high-quality reconstructions. Finally, the differentiable nature of NeRFs makes them compatible with gradient-based optimization, facilitating downstream tasks such as 3D reconstruction, scene editing, or integration into larger learning frameworks.

The increasing adoption of NeRFs and their appealing characteristics motivate our research question: *Is it possible to build an MLLM able to directly ingest NeRFs?* Inspired by recent studies on meta-networks that can process neural fields [102, 69], we answer this question positively by showing that it is possible to process the weights of a given NeRF with a meta-network encoder that projects the NeRF weights into the embedding space of a pre-trained LLM such as LLaMA 2 [122]. By doing so, we create the first MLLM for NeRFs, dubbed Large Language and NeRF Assistant (*LLaNA*), which can perform NeRF-language tasks such as NeRF captioning, NeRF Q&A, and zero-shot NeRF classification.

We also introduce the first NeRF-language benchmark for NeRF textual assistants. This benchmark consists of two datasets of NeRFs paired with language annotations: ShapeNeRF-Text, including 40K synthetic objects from ShapeNet [17] and ObjNeRF-Text, featuring 280K NeRFs trained on synthetic and real 3D models from Objaverse [30].

A natural baseline for building an assistant for NeRFs is to convert the representation into conventional modalities—by rendering images or extracting 3D point clouds—and then process them with existing MLLMs designed for such inputs. We therefore compare *LLaNA* against these indirect approaches on the proposed benchmark. Our results show that the quality of the MLLM’s output is sensitive to the resolution of the extracted geometry or images, and, in the case of images, also to the chosen rendering viewpoints, which can substantially limit performance. Important details may be lost when rendering from suboptimal viewpoints or when the extracted geometry lacks sufficient resolution. In contrast, by operating directly on the MLP weights, our method leverages the full information encoded in the NeRF without requiring additional design choices. As a result, *LLaNA* provides the most effective way

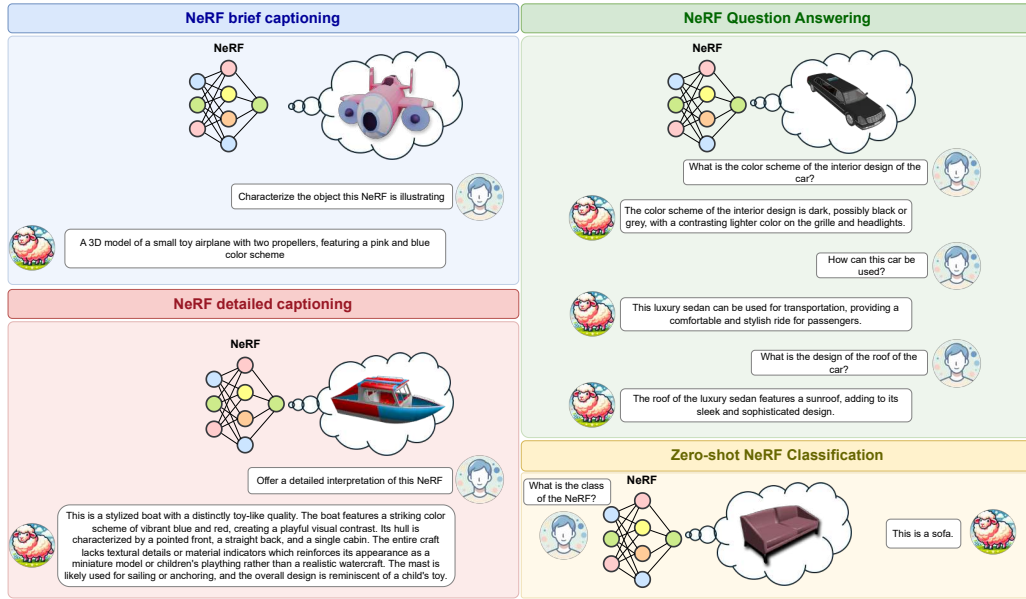


FIGURE 3.1: *LLaNA*. A new Multimodal Large Language Model that understands and reasons on an input NeRF. Notably, our framework processes directly the NeRF weights and performs tasks such as captioning, Q&A, and zero-shot classification of NeRFs.

to build a NeRF assistant, consistently outperforming MLLMs that rely on rendered images or queried 3D geometries.

In this Chapter, the following contributions are presented:

- *LLaNA*, the first MLLM capable of performing tasks such as captioning and Q&A on NeRFs.
- We show that it is possible to build such an assistant by directly processing the NeRFs weights with a meta-encoder, which is faster and captures more information compared to rendering images or extracting 3D data.
- A NeRF-language benchmark for MLLMs, built on ShapeNet and Objaverse, which contains more than 320K NeRFs of synthetic and real objects, paired with text annotations. Our evaluation on this benchmark demonstrates that *LLaNA* outperforms traditional MLLMs operating on discrete representations derived from NeRFs.

3.2 Related Work

Multimodal Large Language Models. Significant advancements in Large Language Models (LLMs) have revolutionized language understanding, reasoning, and generalization [33, 100, 1, 122]. These capabilities have recently

been extended to Multimodal Large Language Models (MLLMs), which integrate sensory modalities such as images [74], audio [44], and video [142].

A common paradigm in MLLM design involves aligning visual features with a textual embedding space to leverage the reasoning power of a frozen LLM. For instance, BLIP-2 [63] introduces a *Querying Transformer* (Q-Former) to bridge the modality gap. This lightweight module uses a set of learnable query tokens to extract a fixed number of relevant visual features from a frozen image encoder, effectively acting as an information bottleneck before passing tokens to the LLM. In contrast, LLaVA [74] adopts a simpler yet effective approach by using a linear projection layer (later upgraded to a Multi-Layer Perceptron in LLaVA-1.5) to map features from a CLIP vision encoder directly into the word embedding space of the pre-trained LLM. While models like Kosmos [45] are trained from scratch, the trend has shifted toward these modular architectures (e.g., InstructBLIP [28], Qwen-VL [9]) that preserve the pre-trained knowledge of the underlying language model, by fine-tuning it.

The extension of MLLMs to 3D data has primarily been driven by two approaches. The first approach is based on the abundance of 2D vision-language data by rendering 3D assets into multi-view images. A prominent example is 3D-LLM [43], which utilizes a 3D feature extractor to reconstruct 3D features from multi-view rendered images, effectively "injecting" 3D spatial information into a pre-trained Vision-Language Model (VLM). It further introduces a 3D localization mechanism with position embeddings to help the model reason about spatial relationships and navigation.

The second approach involves training models directly on native 3D representations, such as point clouds. PointLLM [133] employs a point cloud encoder (Point-BERT [136]) to convert colored point clouds into tokens that the LLM can process. It utilizes a two-stage training strategy: latent space alignment, followed by instruction tuning on a large-scale dataset of point-text pairs. Similarly, GPT4Point [98] introduces a unified framework for both understanding and generation. It utilizes a *Point-Q-Former* for alignment and is supported by *Pyramid-XL*, a large-scale point-language dataset. Beyond simple Q&A, GPT4Point is capable of controllable 3D generation by using the aligned features to guide a diffusion-based 3D generator.

Neural radiance fields. NeRF [88] have been applied in several visual tasks such as novel view synthesis [83], generative media [95], robotics [135], and computational photography [87]. The base formulation employs MLPs to convert spatial coordinates into colors and densities. Recent advancements substitute or enhance MLPs with explicit data structures [18, 117, 35, 90] for

faster training and inference.

Neural radiance fields and language. The interaction between NeRF and language has been recently investigated for several practical applications. Many works address the problem of generating geometrically consistent views of objects or scenes described by textual prompts [111, 85, 51, 110, 62, 60, 95]. Other approaches focus on editing the scene represented by a NeRF through text, e.g., by changing the appearance and shape of objects [127, 46, 114, 128, 118, 39, 137, 145], or by inserting/removing objects in the scene [8, 89]. Some techniques investigate new types of radiance fields that predict language features for each spatial location alongside density and color [54, 57]. By transferring knowledge from vision-language models into these enhanced radiance fields, they can be queried by textual prompts. Such *language fields* are parametrized by a neural network. Unlike all previous methods, the solution proposed in [10] is the first to utilize the weights of a NeRF’s MLP as an input modality. This method aims to learn a mapping between the embedding spaces of the NeRF and CLIP [99] to perform tasks such as NeRF retrieval from textual or image queries. Differently, our goal is to develop an MLLM capable of reasoning about NeRFs.

Deep learning on neural networks. Several studies have explored using meta-networks, i.e., neural networks that process other neural networks. Initially, researchers concentrated on predicting network characteristics, such as accuracy and hyperparameters, by processing their weights [125, 109, 56, 49, 79]. Several recent works focus on processing networks that implicitly encode data, e.g., Implicit Neural Representations (INR) or Neural Fields. These methods are able to classify or segment data by processing solely the weights of the input neural networks. Functa [34] trains a single shared network on a full dataset to learn compact modulation embeddings for each sample, which can then be used for various downstream tasks. More recent research has shifted focus to analyzing networks that represent individual data samples, such as networks trained to model specific objects. By leveraging a novel encoder architecture for MLP weights, *inr2vec* [29] extracts compact embeddings from INRs of 3D shapes, which are employed as inputs for downstream tasks. *nf2vec* [102] extends *inr2vec* to ingest the weights to classify, segment, or retrieve similar NeRFs. The solution from [16] develops a strategy to process neural fields represented by a hybrid tri-plane structure. Other approaches [91, 144, 143] develop equivariant architectures to handle MLPs by exploiting weight space symmetries [40] as an inductive bias. Also, Graph Neural Networks have been investigated to compute a network

representation [58, 69]. Since we aim to process NeRFs directly from the network weights, we employ `nf2vec` as our meta-encoder due to its efficient and scalable architecture.

3.3 Methodology

This section describes the proposed Large Language and NeRF Assistant (*LLaNA*). We first provide an overview of NeRFs and the meta-encoder that maps NeRF weights into a global embedding. Then, we present the overall *LLaNA* framework and discuss our training protocol.

3.3.1 Neural Radiance Fields

A Neural Radiance Field (NeRF) [88] is a framework that employs coordinate-based neural networks, typically Multi-Layer Perceptrons (MLP), to model 3D scenes or objects. It is trained on a set of images taken from various vantage points. Once trained, a NeRF can be used to perform novel views synthesis, i.e., photorealistic rendering of images from viewpoints unseen at training time.

In its base formulation, the MLP is a function of continuous 3D coordinates $\mathbf{p} = (x, y, z) \in \mathbb{R}^3$, that yields a four-dimensional output, $RGB\sigma \in [0, 1]^4$. This output encodes the *RGB* color and the volume density σ at each 3D location in the scene. The volume density σ can be interpreted as the differential probability of a ray terminating at a point $\mathbf{p}$. After training, a NeRF can be used to render images from any desired viewpoint at arbitrary resolution by querying it for the values of *RGB* and σ at several points along the ray corresponding to each pixel and applying the volumetric rendering equation [88].

In this work, we implement NeRFs as MLPs composed of one input layer, L hidden layers, and one output layer. An example of MLP with 1 input, 1 output, and 1 hidden layer is shown in Figure 3.3 (left).

A layer is parameterized by a weight matrix plus a bias vector. More in detail, the hidden layers in our architecture have the same number of input and output neurons, H , thus having squared weight matrices $\mathbf{W}_l \in \mathbb{R}^{H \times H}$ for $l = 1, \dots, L$ and H -dimensional biases $\mathbf{b}_l \in \mathbb{R}^H$. The input $\mathbf{p}$ goes through a 24-frequency encoding [88], therefore the first layer has $\mathbf{W}_{in} \in \mathbb{R}^{144 \times H}$ and $\mathbf{b}_{in} \in \mathbb{R}^H$. The final layer has $\mathbf{W}_{out} \in \mathbb{R}^{H \times 4}$ and $\mathbf{b}_{out} \in \mathbb{R}^4$. Adopting the same architecture used in [102], an instance of the employed NeRF has $L = 3$ hidden layers, with 64 neurons each. The *ReLU* activation function is applied

between all layers except for the last one, which directly computes the density and *RGB* values without any activation function. Our NeRFs are trained using a Smooth L_1 loss [38] between the predicted and ground truth *RGB* pixel intensities, weighting background pixels less than foreground pixels (0.8 foreground vs. 0.2 background). The final rendered images are obtained by querying the neural network with 3D coordinates to obtain *RGB* color values and density estimates. These values are then integrated along camera rays using volumetric rendering techniques [88], to produce the final image. Each NeRF is trained for approximately 2000 steps, until it achieves good reconstruction quality as measured by the Peak Signal-to-Noise Ratio (PSNR).

3.3.2 Meta-encoder

In this work, we investigate how to design a Multimodal Large Language Model (MLLM) that works directly on the weights of NeRFs. We expect the NeRF weights to contain comprehensive information about the represented object, such as its geometry and appearance. Thus, an encoder can extract all the relevant information from these weights to perform language-based tasks like generating captions and answering questions about the object.

Inspired by the recent development of meta-networks capable of processing neural fields [69, 102], we employ *nf2vec* [102] as our meta-encoder architecture. This framework was specifically designed by Ballerini et al. to derive a meaningful and compact representation of an input NeRF solely by processing its weights, without the need to sample the underlying function. A significant challenge in designing such an encoder is scaling to large NeRFs; a naive MLP mapping flattened weights to an embedding could require hundreds of millions of parameters for even small NeRFs. To address this, *nf2vec* treats the weights as a structured input. In particular, the weight matrices and biases of the input NeRF are stacked along the row dimension to form a matrix $\mathbf{M} \in \mathbb{R}^{S \times H}$, where $S = (I+1) + L(H+1) + (H+1)$. In our model $I = 75$, $L = 3$, $H = 64$, then $S = 336$. Figure 3.2 shows the architecture of *nf2vec*.

The meta-encoder is parametrized as an MLP with batch normalization layers [47] and *ReLU* non-linearities. This architecture processes each row of $\mathbf{M}$ independently, which allows it scale gracefully regardless of the depth of the NeRF or the total of parameters. After extracting S tokens of length G , they are processed by a max-pooling layer to provide a global representation

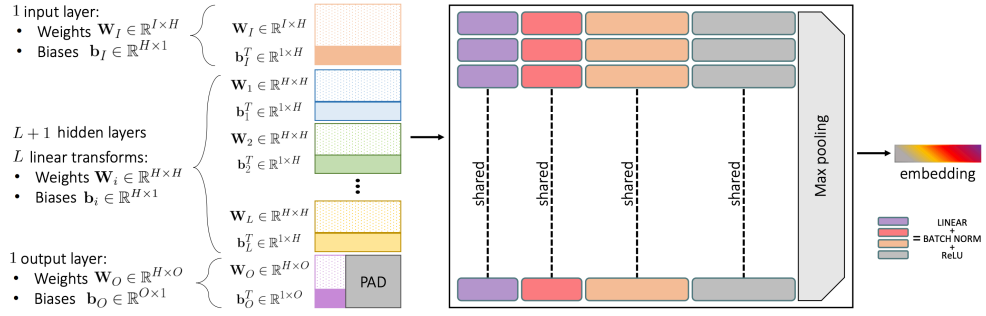


FIGURE 3.2: Architecture of nf2vec.

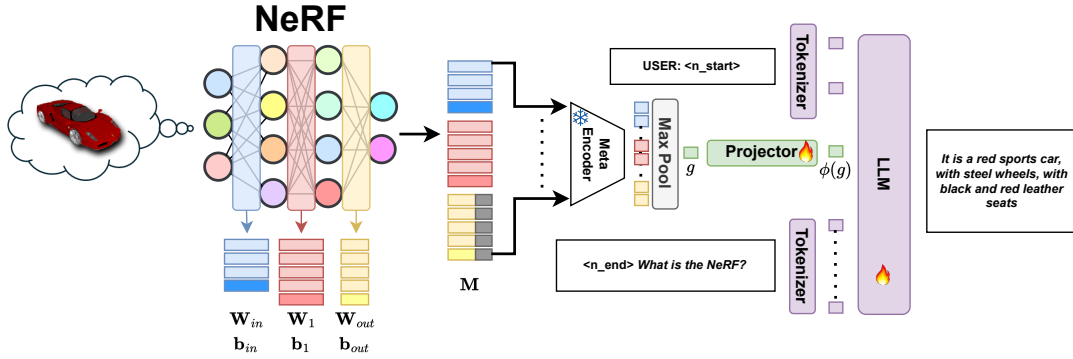


FIGURE 3.3: Framework overview. Example of NeRF captioning.

The meta-encoder of this model is nf2vec, shown in Figure 3.2

$g \in \mathbb{R}^G$ of the NeRF, with $G = 1024$ in our experiments. Crucially, the success of this weight-processing approach relies on the alignment of weights across different NeRFs. Since fitting an individual NeRF involves randomness (e.g., initialization and shuffling), weights in the same position may not share the same semantic role across samples. To ensure a well-organized latent space, we follow the practice of initializing all input NeRFs with a shared, pre-computed random vector, which favors weight alignment and enables framework convergence. The encoder is pre-trained using a self-supervised protocol. It is paired with an implicit decoder that, given the global embedding and a spatial coordinate, predicts the original field values (e.g., RGB values and density σ). The framework is supervised via volumetric rendering, minimizing the regression loss between predicted and true pixel values from arbitrary viewpoints.

This ensures that the resulting embeddings capture a high-fidelity, 3D-consistent representation of the geometry and appearance of the input object.

3.3.3 Large Language and NeRF Assistant

Inspired by recent approaches that proposed effective Multimodal Large Language Models, we build *LLaNA* by leveraging on a pre-trained LLM with a Transformer backbone [126], in our experiments LLaMA 2 [122], and projecting the NeRF modality into its embedding input space, as proposed for images and 3D data [74, 133] (see Figure 3.3, right). Thanks to the self-attention mechanism, the transformer can understand the contextual relationships between the text and the NeRF tokens, enabling it to generate responses based on both the text and the NeRF inputs.

We define a projector network, ϕ , composed of a stack of 3 trainable linear layers, interleaved with *GeLU* activation functions, that project the embedding of the input NeRF computed by the meta-encoder into the embedding space of LLaMA 2. More in detail, the NeRF embedding is encapsulated between two special tokens, $\langle \text{n_start} \rangle$ and $\langle \text{n_end} \rangle$, whose embeddings are learned end-to-end while training. Finally, an input sequence composed by the NeRF embedding and k word tokens, $(\langle \text{n_start} \rangle, \phi(g), \langle \text{n_end} \rangle, w_1, w_2, \dots, w_k)$, is provided as input to the LLM which predicts a sequence of word tokens $(\hat{w}_{k+1}, \hat{w}_{k+2}, \dots, \hat{w}_{eos})$.

3.3.4 Training protocol

To train our framework, we hold multiple conversations about each NeRF leveraging both our ShapeNeRF–Text and ObjaNeRF–Text datasets (see Section 3.4). These conversations are organized into a set of prompts from the user and expected ground truth answers that are used to optimize the original auto-regressive objective of the LLM. For the meta-encoder, we train *nf2vec* on ShapeNeRF–Text and ObjaNeRF–Text for 100 epochs on 4 NVIDIA A100 GPUs.: When training *LLaNA* we follow a two-stage training protocol, keeping the *nf2vec* encoder frozen:

Stage 1: projector training. In the first stage, we train the projector network ϕ to align the NeRF and the word embedding spaces while keeping the LLM weights fixed. We train on an instruction dataset of brief descriptions from ShapeNeRF–Text and ObjaNeRF–Text to learn the projection layer efficiently. We also train the embeddings of the special tokens used to encapsulate the NeRF one. We optimize the projector weights and the embeddings for 3 epochs with a learning rate of 0.002 and a batch size of 16 on each GPU.

Stage 2: instruction tuning. The second stage of training focuses on teaching the model to understand and reason about NeRF data using three types of

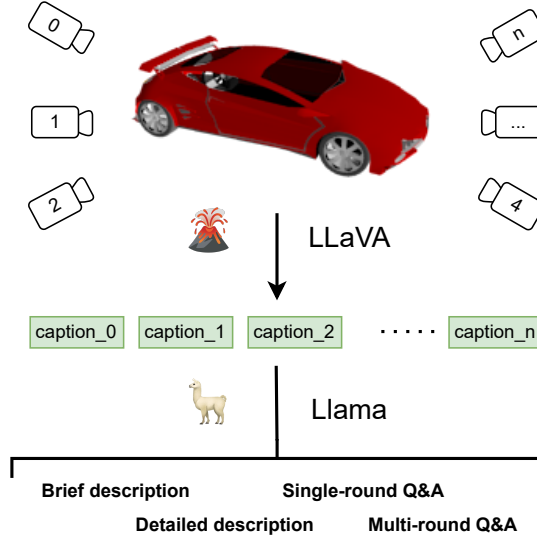


FIGURE 3.4: Automatic annotation pipeline

text from ShapeNeRF-Text and ObjaNeRF-Text: brief descriptions, detailed descriptions, and Q&A conversations. In this phase, we optimize both the projector and the LLM for 3 epochs. We employ a learning rate of 0.00002 and a batch size of 4 on each GPU.

Our model is implemented in PyTorch and trained on NVIDIA A100 GPUs with 64GB of VRAM each. The model variant based on the 7B LLAMA architecture requires 4 GPUs for training, while our largest version, which uses the 13B LLAMA architecture, needs 8 GPUs. Training either version of the model takes approximately one day to complete.

3.4 Benchmark

3.4.1 ShapeNeRF-Text

ShapeNeRF-Text features paired NeRFs and language annotations for ShapeNet objects [17], in particular for all the 40K NeRFs available in the nf2vec dataset [102]. We followed the strategy defined in PointLLM [133] to build the textual annotations. More in detail, we generated a *brief description*, a *detailed description*, 3 *single-round Q&As*, and one *multi-round Q&A* for each object. The brief descriptions are concise captions of the object, taking into account its global structure and appearance. The detailed descriptions are longer sentences that describe all the details of the object. The single-round Q&As consist of a question about the object and the corresponding ground truth

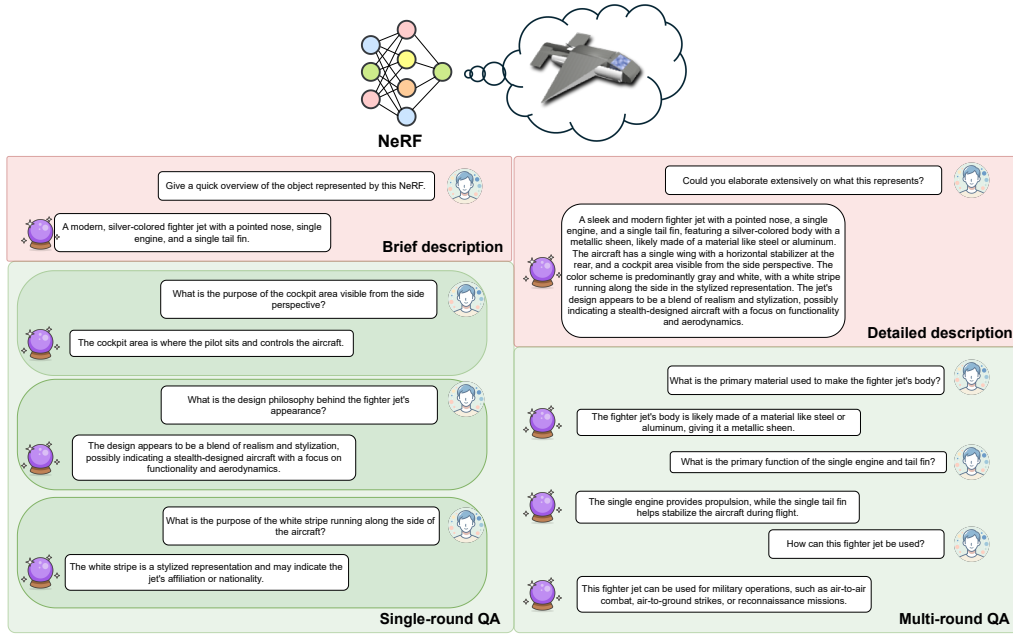


FIGURE 3.5: Example of data sample from ShapeNeRF-Text

answer. Finally, the multi-round Q&As are longer conversations formed by 3 questions and the relative answers. The automatic data annotation pipeline is inspired by Cap3D [80] and is shown in Figure 3.4. First, multiple views of each ShapeNet object have been rendered from different perspectives. Then, each view has been provided as input to LLaVA (LLaVA2-13b) [74] to get a detailed description of the object from that point of view. Afterward, starting from the captions generated by LLaVA, LLaMA 3 (LLaMA3-8B-chat) was used to generate the final ground truth text data (brief and detailed descriptions, single and multi-round Q&As). Both the frozen LLMs employed to create our benchmark (LLaVA2-13b, LLaMA3-8b-chat) are equipped with safeguards.

When building the ground truth data, to ensure diversity in the language annotations, each brief and detailed description has been associated with a question randomly sampled from 30 instructions for each kind of description. Such instructions, together with the carefully engineered request prompts for LLaVA and LLaMA, are reported in our publication [4]. ShapeNeRF-Text comprises 40K objects, with 240K textual annotations: 40K brief captions, 40K detailed descriptions, 120K single-round Q&A, and 40K three-round Q&A. Figure 3.5 shows an example of the resulting textual annotations.

We split the dataset into train, validation, and test sets, stratified on the ShapeNet classes, containing distinct objects in proportions of 80%, 10%, and 10% of the total number of shapes, respectively.

3.4.2 ObjaNeRF-Text

ObjaNeRF-Text was built leveraging on the rendered views from G-Buffer Objaverse [146]. This dataset provides high-quality rendered views of 280K models from Objaverse. Each object was captured from 40 different camera positions: 38 views taken around the object at two different elevation angles, plus one view from above and one from below. We trained a NeRF for each object, using these rendered views. The textual annotations of ObjaNeRF-Text are derived from the dataset proposed in [133]. This work provides machine-generated textual annotations for Objaverse models, divided into the same categories as ShapeNeRF-Text: brief descriptions, detailed descriptions and Q&A conversations. When building ObjaNeRF-Text, we identified the intersection between the 280K objects in G-Buffer Objaverse and the 3D models from this textually annotated dataset. Overall, the training set of ObjaNeRF-Text comprises around 280K 3D models with brief descriptions, and 30K complex text annotations including detailed descriptions and Q&A conversations, associated with 7K different objects. For our evaluation benchmark, we created two different test sets: one for comparison with PointLLM [133] and another for comparison with GPT4Point [98], leveraging the splits proposed in these works. This dualsplit approach was necessary to ensure the largest and most fair evaluation possible since many samples in the test set of PointLLM were used to train GPT4Point. The final test sets contain human-annotated captions for 1366 objects in the PointLLM test set and 518 objects in the GPT4Point test set. To facilitate the training and test of *LLaNA*, the words of the original text annotations referring to the point cloud data structure, such as “point cloud”, “3D point cloud” or “cloud of points” have been modified into “NeRF”.

3.4.3 Language tasks and metrics

The experimental results on ShapeNeRF-Text and ObjaNeRF-Text are reported divided by tasks: brief captioning, detailed captioning and single-round Q&A. Since the test sets of ObjaNeRF-Text contain short human-annotated captions, such results fall into the brief captioning category. Furthermore, for this task, we also evaluate the methods on the GPT2Shape Human Shape Text (HST) dataset [5], a subset of ShapeNet for which human-curated brief descriptions are publicly available. We employ standard language similarity metrics to evaluate these methods. We compute the cosine similarity between the global embeddings of the generated and ground truth sentences

provided by the pre-trained encoders Sentence-BERT [103] and SimCSE [37]. These metrics were shown to be the most effective in measuring the quality of the generated output [133]. We also include standard handcrafted metrics based on n-gram statistics, like BLEU-1 [93], ROUGE-L [71], and METEOR [11].

3.5 Experimental results

As our method is the first to investigate language tasks on NeRF, there are no baselines in the literature. However, given a NeRF, a straightforward way to create an assistant for it could be to render an image and use an MLLM capable of ingesting images. Alternatively, we could extract the 3D shape from the NeRF and use one of the recent 3D MLLMs. Hence, we evaluate LLaVA (v1.6) [74] and BLIP-2 [63] for images, as well as PointLLM [133] and GPT4Point [98] for colored point clouds. Since NeRFs can render arbitrary viewpoints after training, we also include the evaluation of LLaVA [74] in a multi-view scenario. More in detail, we render images from N viewpoints randomly sampled between the set of camera poses used to train each NeRF; then, we concatenate tokens from these N images and fed them into LLaVA alongside text instructions. We set $N=3$ because the model cannot process a higher number of images correctly. In addition, we test 3D-LLM [43], which processes meshes and multi-view images. When evaluating the baselines on ShapeNeRF-Text and ObjNeRF-Text, we employ the official code and pre-trained models released by the respective authors¹.

3.5.1 Experiments on ShapeNeRF-Text and ObjNeRF-Text

Brief captioning The results for the brief description task are reported in Tables 3.1, 3.2, 3.3, and 3.4. When evaluating image-based methods on ObjNeRF-Text, we use random object views since Objaverse 3D models lack consistent orientation, making it impossible to identify standard front and back views as in ShapeNeRF-Text.

LLaNA-13b consistently achieves the best performance across all metrics, often with substantial margins over the runner-up. Notably, the second-best

¹LLaVA: <https://github.com/haotian-liu/LLaVA> BLIP-2: <https://github.com/salesforce/LAVIS/tree/main/projects/blip2> PointLLM: <https://github.com/OpenRobotLab/PointLLM> GPT4Point: <https://github.com/Pointcept/GPT4Point> 3D-LLM: <https://github.com/UMass-Foundation-Model/3D-LLM>

Model	Modality	S-BERT	SimCSE	BLEU-1	ROUGE-L	METEOR
GPT4Point-Opt-2.7b	Point cloud	41.85	40.22	11.76	16.54	11.63
3D-LLM FlanT5-xl	Mesh + MV	59.46	56.42	12.69	21.49	14.32
LLaVA-vicuna-7b	Image (FV)	<u>59.85</u>	<u>62.35</u>	22.67	<u>23.24</u>	<u>23.35</u>
LLaVA-vicuna-7b	Image (BV)	55.68	58.46	21.97	22.46	22.50
LLaVA-vicuna-7b	Image (MV)	59.77	61.42	<u>23.16</u>	22.68	23.01
PointLLM-7b	Point cloud	49.59	48.84	16.74	17.92	14.56
LLaNA-7b	NeRF	74.94	76.41	42.73	43.64	41.95
BLIP-2 FlanT5-xxl	Image (FV)	56.13	58.21	5.46	18.69	9.67
BLIP-2 FlanT5-xxl	Image (BV)	52.48	54.05	5.67	18.20	9.50
LLaVA-vicuna-13b	Image (FV)	<u>61.00</u>	<u>61.16</u>	14.30	20.00	<u>23.31</u>
LLaVA-vicuna-13b	Image (BV)	54.35	56.09	21.94	21.67	22.09
LLaVA-vicuna-13b	Image (MV)	59.64	61.01	<u>22.84</u>	<u>22.17</u>	23.08
PointLLM-13b	Point cloud	51.54	50.35	17.20	18.51	14.92
LLaNA-13b	NeRF	75.09	76.45	42.83	43.70	42.21

TABLE 3.1: **NeRF brief captioning** results on ShapeNeRF-Text. Best results are in **bold**, runner-up is underlined.

Model	Modality	S-BERT	SimCSE	BLEU-1	ROUGE-L	METEOR
GPT4Point-Opt-2.7b	Point cloud	43.15	42.22	12.02	18.73	13.69
3D-LLM FlanT5-xl	Mesh + MV	<u>56.07</u>	52.13	<u>15.94</u>	<u>20.71</u>	<u>15.22</u>
LLaVA-vicuna-7b	Image (FV)	54.31	56.28	10.08	14.71	14.53
LLaVA-vicuna-7b	Image (BV)	51.75	52.29	8.13	13.96	14.18
LLaVA-vicuna-7b	Image (MV)	54.15	<u>56.87</u>	10.32	12.76	15.13
PointLLM-7b	Point cloud	43.40	44.50	8.53	11.64	9.97
LLaNA-7b	NeRF	64.78	65.01	20.65	23.24	24.10
BLIP-2 FlanT5-xxl	Image (FV)	<u>57.11</u>	<u>59.43</u>	8.21	<u>18.02</u>	12.14
BLIP-2 FlanT5-xxl	Image (BV)	54.11	56.37	9.09	17.38	11.79
LLaVA-vicuna-13b	Image (FV)	55.62	55.56	6.56	11.81	14.52
LLaVA-vicuna-13b	Image (BV)	50.00	50.79	9.39	12.76	14.46
LLaVA-vicuna-13b	Image (MV)	54.25	55.56	<u>9.78</u>	14.13	<u>14.99</u>
PointLLM-13b	Point cloud	45.41	46.39	9.57	12.38	11.92
LLaNA-13b	NeRF	65.66	66.09	20.80	24.28	25.90

TABLE 3.2: **NeRF brief captioning** results on HST. Best results are in **bold**, runner-up is underlined.

Model	Modality	S-BERT	SimCSE	BLEU-1	ROUGE-L	METEOR
3D-LLM FlanT5-xl	Mesh + MV	38.22	39.60	5.47	7.29	10.75
LLaVA-vicuna-7b	Image (RV)	39.29	40.93	5.22	8.35	12.49
LLaVA-vicuna-7b	Image (MV)	<u>40.15</u>	<u>41.09</u>	<u>5.62</u>	<u>9.07</u>	<u>13.22</u>
PointLLM-7b	Point cloud	38.81	40.11	5.55	8.39	11.24
LLaNA-7b	NeRF	41.36	42.28	13.21	16.93	15.63
BLIP-2 FlanT5-xxl	Image (RV)	35.48	35.19	<u>7.69</u>	<u>13.75</u>	11.68
LLaVA-vicuna-13b	Image (RV)	38.57	39.05	4.62	<u>7.63</u>	11.85
LLaVA-vicuna-13b	Image (MV)	<u>41.01</u>	<u>41.10</u>	4.87	8.03	<u>12.35</u>
PointLLM-13b	Point cloud	39.64	40.63	5.94	8.43	11.63
LLaNA-13b	NeRF	42.08	42.40	13.86	17.51	16.18

TABLE 3.3: **NeRF brief captioning** results on ObjaNeRF-Text (PointLLM test set). Best results are in **bold**, runner-up is underlined.

results are most frequently obtained by *LLaNA-7b*, which shares the same architecture but employs a smaller LLM.

The difference in the quality of the captions generated by *LLaNA* compared to the baselines is shown in Figures 3.6, 3.7, and 3.8, where the descriptions provided by *LLaNA* are the most accurate.

A consistent trend observed in both the tables and qualitative results is that image-based models generally outperform those processing point clouds.

Model	Modality	S-BERT	SimCSE	BLEU-1	ROUGE-L	METEOR
GPT4Point-Opt-2.7b	Point cloud	34.42	31.89	8.76	<u>13.62</u>	13.55
3D-LLM FlanT5-xl	Mesh + MV	41.26	37.74	<u>11.08</u>	12.26	15.58
LLaVA-vicuna-7b	Image (RV)	41.59	42.38	7.24	11.74	16.01
LLaVA-vicuna-7b	Image (RV)	<u>42.34</u>	<u>43.21</u>	8.59	11.94	<u>18.52</u>
LLaNA-7b	NeRF	43.73	43.09	20.22	25.15	22.21
BLIP-2 FlanT5-xxl	Image (RV)	37.24	37.32	<u>10.63</u>	<u>16.87</u>	14.69
LLaVA-vicuna-13b	Image (RV)	42.08	41.04	6.53	10.57	15.67
LLaVA-vicuna-13b	Image (MV)	<u>44.15</u>	<u>43.19</u>	6.75	10.86	<u>16.10</u>
LLaNA-13b	NeRF	44.26	43.75	20.61	25.62	22.36

TABLE 3.4: **NeRF brief captioning** results on ObjaNeRF-Text (GPT4Point test set). Best results are in **bold**, runner-up is underlined.

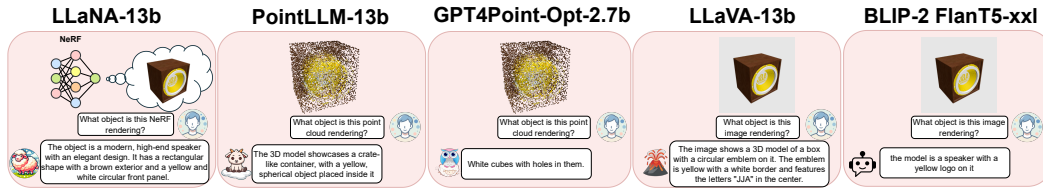


FIGURE 3.6: Qualitative results on ShapeNeRF-Text brief descriptions.

This is likely attributable to the larger training datasets used for the image modality, millions of images compared to only hundreds of thousands of shapes, which enhances generalization. Additionally, images can capture more fine-grained details than point clouds at the input resolutions required by image-based versus 3D MLLMs. Nonetheless, our method, which operates on NeRFs, benefits from a holistic view of the object and provides the most accurate descriptions. Remarkably, in *LLaNA*, all the necessary information for this language task can be extracted from a single global embedding obtained by directly processing the NeRF weights. Comparing the results of image-based MLLMs when processing front versus back views, we can see that the vantage point has a non-negligible effect on the performance of such baselines, with SentenceBERT and SimCSE metrics diminishing by about 4 points in all baselines. In a dataset without canonical poses for objects, this would be a relevant limitation that processing NeRF weights seamlessly sidesteps. Finally, we observe that the multi-view setup of LLaVA provides better performance to the single-view counterpart. Regarding LLM size scaling, our analysis reveals only marginal performance improvements with larger models in brief captioning tasks. On ShapeNeRF-Text, LLaVA shows a minimal improvement in S-BERT scores from 59.85 to 61.00 when using the 13B parameter model, while other metrics actually deteriorate. Similarly, PointLLM exhibits modest gains of approximately 1 point across metrics, and LLaNA demonstrates even smaller improvements. This trend is consistent across HST and ObjaNeRF-Text datasets. These findings suggest that the ability

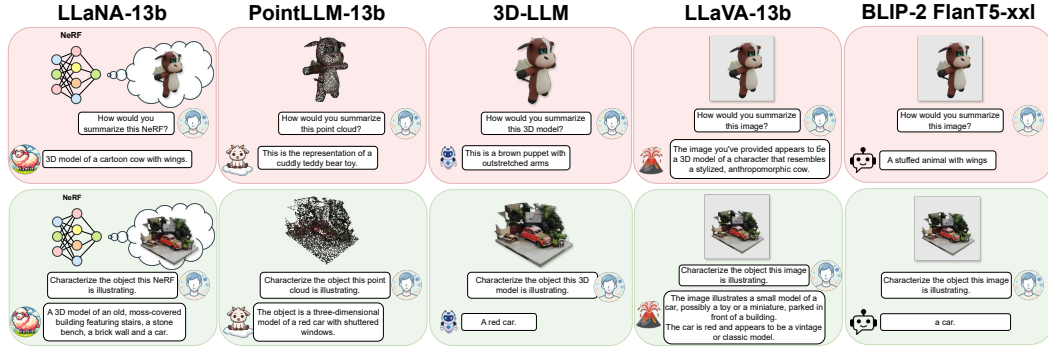


FIGURE 3.7: Qualitative results on ObjNeRF-Text brief descriptions (PointLLM test set).

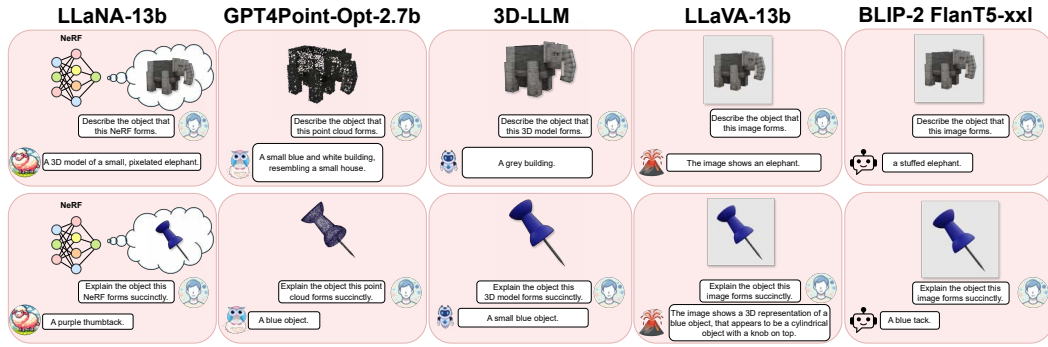


FIGURE 3.8: Qualitative results on ObjNeRF-Text brief descriptions (GPT4Point test set).

to perform NeRF-language tasks is not strongly correlated with LLM size. Unlike traditional NLP tasks, where larger models generally lead to significant performance boosts due to their enhanced language understanding and generation capabilities, NeRF-based captioning appears to depend more on the model’s ability to process and integrate 3D and visual information from the input NeRF. Given the substantial computational costs of training and deploying larger models, the minimal performance gains observed in brief captioning tasks may not justify the increased resource demands. Another key observation is that the quality of the input encoding and the processing pipeline, which turns NeRF representations into LLM-compatible features, have a greater impact on performance than increasing the size of the underlying LLM. For example, despite using LLMs of similar sizes (2.7B for GPT4Point and 3B for 3D-LLM), these models exhibit very different performance levels. One possible explanation is that these tasks require precise spatial and geometric reasoning, which may not inherently improve with a larger LLM. Models like 3D-LLM, which incorporate multi-view processing and colored meshes, likely benefit more from their specialized architecture

Model	Modality	S-BERT	SimCSE	BLEU-1	ROUGE-L	METEOR
GPT4Point-Opt-2.7b	Point cloud	42.44	38.33	3.72	9.21	5.13
3D-LLM FlanT5-xl	Mesh + MV	58.00	53.91	1.58	14.40	5.28
LLaVA-vicuna-7b	Image (FV)	57.55	57.68	14.99	22.82	14.36
LLaVA-vicuna-7b	Image (BV)	53.11	54.46	14.73	22.47	14.05
LLaVA-vicuna-7b	Image (MV)	55.26	<u>58.46</u>	<u>15.07</u>	<u>24.05</u>	<u>14.85</u>
PointLLM-7b	Point cloud	<u>59.02</u>	58.30	10.28	19.26	10.55
LLaNA-7b	NeRF	75.25	77.42	19.57	32.96	20.45
BLIP-2 FlanT5-xxl	Image (FV)	41.27	40.69	0.18	7.83	2.60
BLIP-2 FlanT5-xxl	Image (BV)	38.49	37.89	0.19	7.72	2.58
LLaVA-vicuna-13b	Image (FV)	59.08	58.87	23.63	23.55	22.55
LLaVA-vicuna-13b	Image (BV)	50.09	50.33	13.77	21.36	13.18
LLaVA-vicuna-13b	Image (MV)	<u>60.21</u>	<u>59.51</u>	15.07	<u>32.16</u>	14.64
PointLLM-13b	Point cloud	59.64	58.55	10.52	19.44	10.83
LLaNA-13b	NeRF	75.51	77.63	<u>19.87</u>	32.93	<u>20.46</u>

TABLE 3.5: NeRF detailed captioning on ShapeNeRF-Text. Best results are in **bold**, runner-up is underlined. (FV: front-view, BV: back-view, MV: multi-view)

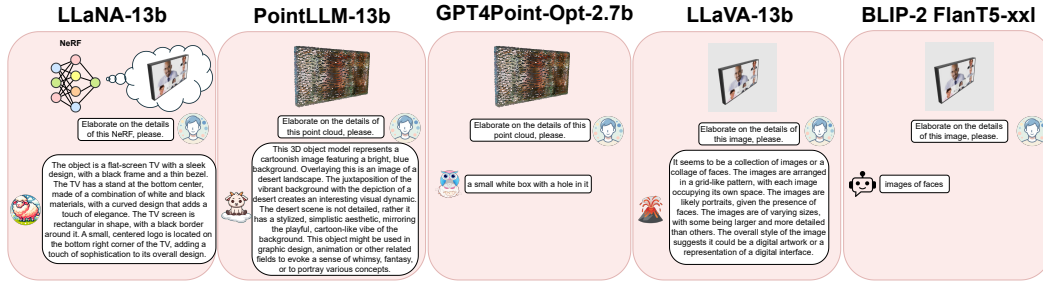


FIGURE 3.9: Qualitative results on ShapeNeRF-Text detailed descriptions. From top to bottom: brief and detailed descriptions, single-round Q&A

than from parameter scaling.

Detailed captioning The results for the detailed captioning task are presented in Table 3.5. *LLaNA* demonstrates superior performance compared to all other models, showing significant improvements in data-driven metrics such as Sentence-BERT and SimCSE. For traditional metrics like BLEU-1, ROUGE-L, and METEOR, our model achieves comparable results to LLaVA. 3D-LLM [43], processing multi-view images and colored meshes, performs well on the Sentence-BERT metric, whereas all other metrics show poor results. Interestingly, the point-based model PointLLM [133] performs similarly to the image based one, LLaVA [74]. Considering the Sentence-BERT metric, *LLaNA*-13b achieves 75.51, notably 15.87 points more than PointLLM and 15.30 points more than LLaVA-13b multi-view setup. These substantial performance gaps suggest that, while individual or aggregated images may be sufficient for brief descriptions, they may lack all the details needed to provide a comprehensive description. Moreover, the dependency of the output quality on the selected vantage points remains strong, as proven by the varying performance

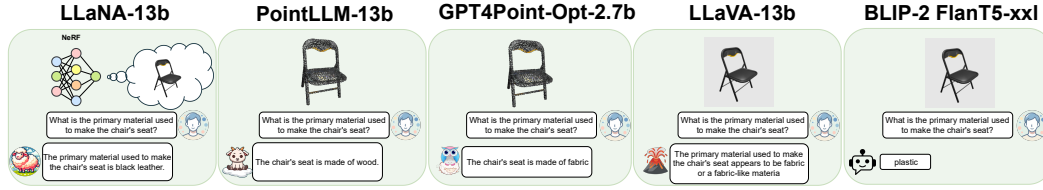


FIGURE 3.10: **Qualitative results on ShapeNeRF-Text single-round Q&A.**

achieved by LLaVA across front-view, back-view, and multi-view scenarios. In contrast, the NeRF weights contain detailed and complete information about the object, which is fundamental for more granular description tasks, with the additional advantage of not requiring tuning such hyperparameters. The ability of NeRF to capture holistic information about the object is also shown in Figure 3.9, where only the direct processing of NeRF weights lets LLaNA understand that the object is a TV. PointLLM and LLaVA generate detailed but wrong descriptions, likely because of the need to extract the intermediate discrete representation as a point cloud or an image, losing information. Indeed, in both cases, it is hard even for a human observer to provide the right description from the intermediate modalities shown in the figure. When comparing the different versions of our model, the 13B variant slightly outperforms its 7B counterpart, a pattern consistently observed across other models in the table, such as PointLLM and LLaVA. Specifically, the relative improvements from 7B to 13B versions on Sentence-BERT and SimCSE are: 0.62 and 0.25 for PointLLM, 1.53 and 1.19 for LLaVA, 0.26 and 0.21 for LLaNA. This limited performance gain suggests that the architectural improvements and training strategies employed in these models may be more crucial for performance than simply scaling up model size, as also observed with the results obtained in the brief captioning task. Regarding the training strategies of these models, an interesting observation can be made. Overall, the highest performance on NeRF detailed captioning is achieved by LLaNA, LLaVA, and PointLLM; notably, the only models incorporating LLM fine-tuning in their training protocol. This pattern strongly suggests that fine-tuning the LLM plays a crucial role in enhancing the quality of the generated descriptions. While this fine-tuning stage appears less critical for generating brief captions, it significantly impacts the quality of longer, detailed descriptions. This can be attributed to the ability of the fine-tuned LLM to adapt its pre-trained language understanding capabilities to the specific characteristics and vocabulary of NeRF-based object descriptions.

Model	Modality	S-BERT	SimCSE	BLEU-1	ROUGE-L	METEOR
GPT4Point-Opt-2.7b	Point cloud	27.62	31.41	6.26	9.38	5.41
3D-LLM FlanT5-xl	Mesh + MV	69.62	67.55	32.19	40.95	35.83
LLaVA-vicuna-7b	Image (FV)	71.79	71.96	25.79	34.04	34.86
LLaVA-vicuna-7b	Image (BV)	70.88	70.93	25.17	33.30	34.22
LLaVA-vicuna-7b	Image (MV)	72.26	70.67	24.09	34.67	35.64
PointLLM-7b	Point cloud	<u>74.70</u>	<u>74.40</u>	<u>36.81</u>	<u>44.41</u>	<u>39.76</u>
<i>LLaNA</i> -7b	NeRF	81.03	81.61	45.98	53.27	49.97
BLIP-2 FlanT5-xxl	Image (FV)	45.20	47.92	11.50	20.16	13.49
BLIP-2 FlanT5-xxl	Image (BV)	45.06	47.66	11.50	19.98	13.44
LLaVA-vicuna-13b	Image (FV)	71.61	70.98	20.19	30.42	32.53
LLaVA-vicuna-13b	Image (BV)	68.25	69.06	20.03	29.84	32.27
LLaVA-vicuna-13b	Image (MV)	71.84	71.16	20.04	30.20	33.46
PointLLM-13b	Point cloud	<u>74.65</u>	<u>74.18</u>	<u>37.16</u>	<u>44.86</u>	<u>40.13</u>
<i>LLaNA</i> -13b	NeRF	81.05	81.60	46.08	53.44	49.98

TABLE 3.6: **NeRF single-round Q&A on ShapeNeRF-Text.**Best results are in **bold**, runner-up is underlined.

(FV: front-view, BV: back-view, MV: multi-view)

Single-round Q&A In the single-round Q&A experiment, we test the ability of the assistants to provide accurate answers to specific questions about the object. We prompt the models with the NeRF, or the image/cloud extracted from it, followed by one of the questions in the single-round Q&A annotations associated with the NeRF. We then collect the answer generated by the model and compare it against the ground truth answer with the selected metrics. Results are reported in Table 3.6. Interestingly, PointLLM [133] performs better than LLaVA [74] in this task, likely because it has been specifically trained to answer detailed questions about objects represented as point clouds. Nevertheless, *LLaNA* maintains its position as the top-performing method across all metrics by substantial margins, mirroring our findings from the brief and detailed captioning tasks. Using the 13B LLAMA backbone, the performance gaps between *LLaNA* and the second-best model, PointLLM, are large: 6.40 for Sentence-BERT, 7.42 for SimCSE, 8.92 for BLEU-1, 8.58 for ROUGE-L, and 9.85 for METEOR. Notably, these margins remain consistently large even when using the 7B LLAMA backbone. The performance advantage of *LLaNA* shows that our meta-encoder and projector architecture are capable of effectively extracting fine-grained information from the NeRF representation, even if they are processing directly NeRF weights. Remarkably, the amount of information they can extract lets *LLaNA* answer more precisely than when images or point clouds are extracted from the NeRF. Indeed, as shown in Figure 3.10 which reports a qualitative example from ShapeNeRF-Text, the only assistant able to answer correctly to a precise question about the material of the chair is *LLaNA*. Similar to our findings with brief and detailed descriptions, the Q&A results show minimal performance difference between *LLaNA*-13B and *LLaNA*-7B, further reinforcing that MLLM performance on

Model	Modality	Accuracy (%)
GPT4Point-Opt-2.7b	Point cloud	41.93
3D-LLM FlanT5-xl	Mesh + MV	60.55
LLaVA-vicuna-7b	Image (FV)	60.25
LLaVA-vicuna-7b	Image (BV)	57.00
LLaVA-vicuna-7b	Image (MV)	<u>67.25</u>
PointLLM-7b	Point cloud	50.14
LLaNA-7b	NeRF	67.56
BLIP-2 FlanT5-xxl	Image (FV)	63.67
BLIP-2 FlanT5-xxl	Image (BV)	61.47
LLaVA-vicuna-13b	Image (FV)	66.13
LLaVA-vicuna-13b	Image (BV)	63.90
LLaVA-vicuna-13b	Image (MV)	73.45
PointLLM-13b	Point cloud	48.72
LLaNA-13b	NeRF	<u>69.27</u>

TABLE 3.7: **Zero-Shot NeRF Classification.**
 Best results are in **bold**, runner-up is underlined.
 (FV: front-view, BV: back-view, MV: multi-view)

these tasks is not strongly dependent on the size of the underlying language model. Furthermore, the consistently superior performance of *LLaNA*, *LLaVA*, and *PointLLM* across both tasks underscores the critical role of LLM finetuning in developing models that can effectively describe and answer questions regarding 3D objects.

Zero-shot classification We compare assistants on the task of zero-shot classification. We query the models with the sentence “What is the class of the NeRF/image/cloud? Choose among these: [ShapeNet classes]” where [ShapeNet classes] are the 10 ShapeNet classes available in ShapeNeRF-Text. We consider the answer correct only if the ground truth class appears in the response. We report results in Table 3.7 on ShapeNeRF-Text. Using multiple views boosts the zero-shot classification performance of *LLaVA*-13b, which turns out to be the best model for this task, followed by *LLaNA*-13b. Similar to the brief captioning task, image-based models tend to outperform point cloud-based models on this classification task. This performance pattern aligns with the requirements of these tasks. For brief captioning and classification, which primarily require high-level understanding and concise outputs, image-based models excel by leveraging visual features directly from 2D views, where the nature of the object and its appearance are readily accessible. However, the pattern reverses for detailed captioning and Q&A tasks, where geometric precision and spatial understanding become crucial. These tasks often require reasoning about specific object parts, their relationships, and fine-grained spatial details - information that is inherently preserved in point cloud representations. While image-based models might struggle with occlusions and

loss of information due to the chosen vantage points, point cloud-based approaches can directly reason about the complete 3D geometry, leading to more accurate and detailed responses.

Model	Modality	S-BERT	SimCSE	BLEU-1	ROUGE-L	METEOR
GPT4Point-Opt-2.7b	Point cloud	37.96	39.00	21.33	22.29	<u>24.88</u>
PointLLM-7b	Point cloud	<u>55.48</u>	<u>57.28</u>	<u>21.67</u>	<u>25.84</u>	24.54
LLaNA-7b	NeRF	76.04	77.13	45.00	45.87	44.52
LLaVA-vicuna-13b	Image (FV)	42.86	43.22	15.56	13.74	15.27
LLaVA-vicuna-13b	Image (MV)	44.29	41.85	16.67	16.26	17.58

TABLE 3.8: **Brief captioning on ShapeNeRF-Text of trained baselines.** Best results are in **bold**, runner-up is underlined.

Model	Modality	S-BERT	SimCSE	BLEU-1	ROUGE-L	METEOR
GPT4Point-Opt-2.7B	Point cloud	30.50	31.08	8.12	12.35	11.62
PointLLM-7b	Point cloud	<u>44.65</u>	<u>44.68</u>	8.91	12.33	<u>12.64</u>
LLaNA-7b	NeRF	65.42	66.44	19.63	23.39	23.46
LLaVA-vicuna-13b	Image (FV)	33.79	42.66	10.28	<u>13.22</u>	12.19
LLaVA-vicuna-13b	Image (MV)	34.45	42.54	<u>11.04</u>	12.36	11.48

TABLE 3.9: **Brief captioning on HST of trained baselines.** Best results are in **bold**, runner-up is underlined.

Model	Modality	S-BERT	SimCSE	BLEU-1	ROUGE-L	METEOR
GPT4Point-Opt-2.7b	Point cloud	41.33	40.52	14.48	19.15	<u>13.80</u>
PointLLM-7b	Point cloud	<u>67.30</u>	<u>59.56</u>	<u>15.39</u>	21.42	11.37
LLaNA-7b	NeRF	75.83	77.92	20.22	33.80	21.20
LLaVA-vicuna-13b	Image (FV)	44.69	42.31	10.08	23.46	12.70
LLaVA-vicuna-13b	Image (MV)	45.72	42.49	12.17	<u>23.48</u>	14.25

TABLE 3.10: **Detailed captioning on ShapeNeRF-Text of trained baselines.** Best results are in **bold**, runner-up is underlined.

Model	Modality	S-BERT	SimCSE	BLEU-1	ROUGE-L	METEOR
GPT4Point-Opt-2.7b	Point cloud	22.22	28.66	8.76	13.46	14.19
PointLLM-7b	Point cloud	<u>79.24</u>	<u>80.38</u>	<u>46.00</u>	<u>52.60</u>	<u>42.36</u>
LLaNA-7b	NeRF	81.94	82.43	47.73	54.97	51.78
LLaVA-vicuna-13b	Image (FV)	56.29	62.36	26.87	29.55	30.49
LLaVA-vicuna-13b	Image (MV)	56.42	64.47	25.41	30.09	32.04

TABLE 3.11: **Single-round Q&A on ShapeNeRF-Text of trained baselines.** Best results are in **bold**, runner-up is underlined.

3.5.2 Ablation study on training data

In this section, we conduct an ablation study where we train from scratch some methods on the same training set, i.e. ShapeNeRF-Text. This is done to assess the influence of the training set on the results. Due to the considerable computational resources required for training these models, we evaluated a

Model	Modality	Accuracy (%)
GPT4Point-Opt-2.7b	Point cloud	26.30
PointLLM-7b	Point cloud	<u>49.69</u>
<i>LLaNA</i> -7b	NeRF	69.06
LLaVA-vicuna-13b	Image (FV)	36.49
LLaVA-vicuna-13b	Image (FV)	39.27

TABLE 3.12: **Zero-shot NeRF classification of baselines trained on ShapeNeRF-Text.** Best results are in **bold**, runner-up is underlined. (FV: front-view, MV: multi-view)

Model	Task	Dataset	SBERT	SimCSE	BLEU-1	ROUGE-L	METEOR
LLaMA-7b	brief	Shape	31.08	27.44	22.13	19.39	17.44
LLaMA-13b	brief	Shape	29.29	24.69	16.95	18.93	13.85
LLaMA-7b	brief	HST	26.05	21.48	13.85	14.74	10.48
LLaMA-13b	brief	HST	26.83	21.44	13.19	15.30	10.12
LLaMA-7b	brief	Obja-P	28.20	29.33	10.37	13.59	12.95
LLaMA-13b	brief	Obja-P	23.19	21.29	10.25	13.45	10.91
LLaMA-7b	brief	Obja-G	23.82	21.70	13.21	16.79	13.50
LLaMA-13b	brief	Obja-G	23.43	19.54	14.18	17.40	13.81
LLaMA-7b	detailed	Shape	37.51	32.47	9.66	16.67	9.67
LLaMA-13b	detailed	Shape	40.20	35.80	11.35	18.82	11.29
LLaMA-7b	Q&A	Shape	76.89	76.61	42.01	49.74	45.69
LLaMA-13b	Q&A	Shape	76.85	76.79	42.30	49.92	45.86

TABLE 3.13: **Language-only baselines NeRF captioning and NeRF single-round Q&A.** (Shape: ShapeNeRF-Text, Obja-P: ObjaNeRF-Text PointLLM test set, Obja-G: ObjaNeRF-Text GPT4Point test set)

subset of baselines using their official training code. Accordingly, we followed their protocol, which, for all methods, keeps the modality-specific encoder frozen and trains an adaptor. In PointLLM and LLaVA, the LLM is finetuned during an additional training stage.

Tables 3.8, 3.9, 3.10, 3.11, and 3.12 report results for both *LLaNA* and the baselines trained solely on ShapeNeRF-Text. We notice that these baselines exhibit different behaviors to their pre-trained counterparts, with LLaVA performing significantly worse and PointLLM showing clear improvements. As for GPT4Point, we observe greater variability across metrics; however, overall, it shows no significant benefit from training on ShapeNeRF-Text. Also in this scenario, *LLaNA* yields the best performance compared to all baselines.

3.5.3 Is the LLM all you need?

Finally, we investigate how LLAMA 2, the LLM on which *LLaNA* relies, performs on NeRF-language tasks. For this evaluation protocol, the LLM, finetuned during the second training stage of *LLaNA*, is provided with questions

belonging to our datasets, requiring it to generate correct answers without access to NeRF data. Consequently, its predictions rely solely on textual patterns present in the training set. These experiments offer valuable insights into the language annotations in ShapeNeRF-Text and ObjaNeRF-Text, as well as the impact of different LLM sizes. Results are shown in Table 3.13. Comparing this table with the ones reporting the results of *LLaNA*, we observe a significant performance gap between *LLaNA* and LLAMA. For instance, using the Sentence-BERT metric, *LLaNA* achieves scores of 75.09 and 75.51 on the brief and detailed captioning tasks of ShapeNeRF-Text, respectively, while LLAMA-13B attains only 29.29 and 40.20. This corresponds to performance drops of approximately 61% and 47%. Similarly, for the brief captioning task on ObjaNeRF-Text, the performance drop is around 45% on the PointLLM test set and 47% on the GPT4Point test set. These substantial gaps indicate that ShapeNeRF-Text and ObjaNeRF-Text provide language tasks that require access to 3D object information. Without the information from the NeRF token, the LLM is not able to provide correct descriptions. Therefore, our datasets can be used as reliable benchmarks for evaluating NeRF-language tasks. With regards to the single-round Q&A annotations, LLAMA-2 can answer correctly to a large set of questions leading to a limited performance gap with *LLaNA*: from 81.05 to 76.85, corresponding to a relative decrease of 5%. This relatively small difference can be attributed to the nature of the questions themselves: many of them rely on common sense and general object knowledge rather than specific 3D object information. For instance, questions like *How can the filing cabinet be used to organize office documents?*, *What is a suitable use for this table?* can be answered without detailed information from the corresponding 3D objects. These types of questions were deliberately included to preserve the strong reasoning capabilities of LLAMA while creating more natural and comprehensive conversations. An additional noteworthy finding concerns the relationship between LLM size and performance on NeRF-language tasks. As discussed previously, using a larger LLM does not significantly improve the performance of *LLaNA*. This pattern is also evident in Table 3.13, where LLAMA-7b and LLAMA-13b provide very similar results on ShapeNeRF-Text and ObjaNeRF-Text. This suggests that the ability of the model to process and comprehend 3D object inputs - through its pre-trained encoder and projection layers - is more important than the size of the LLM.

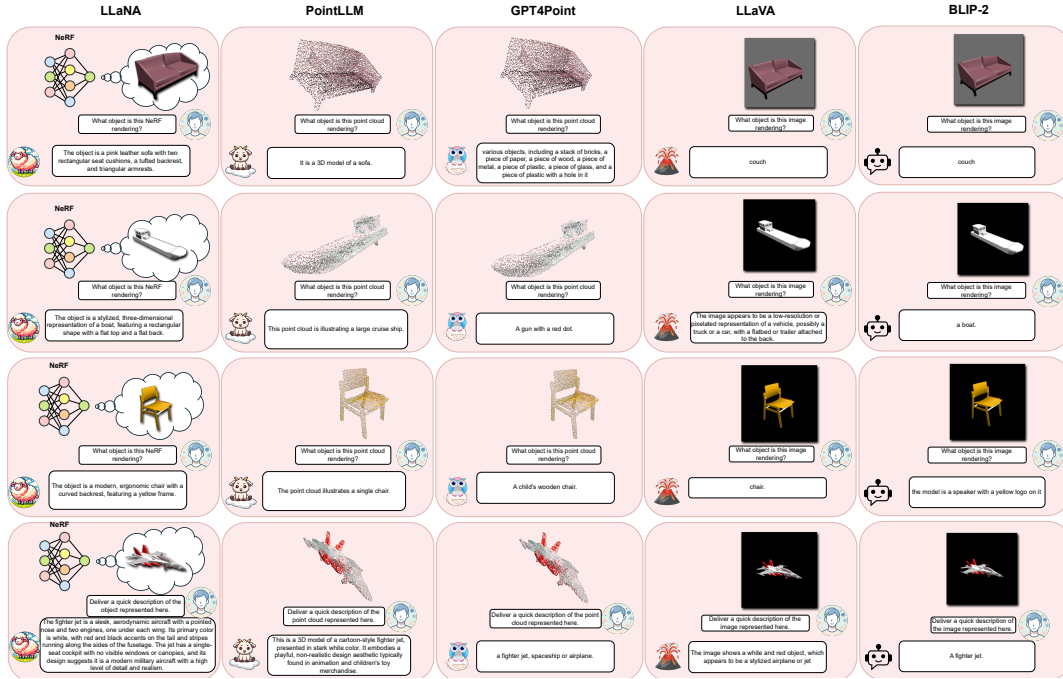


FIGURE 3.11: Brief description example from ShapeNeRF-Text.

3.5.4 Additional qualitative examples

Figures 3.11, 3.12, and 3.13 show additional qualitative examples.

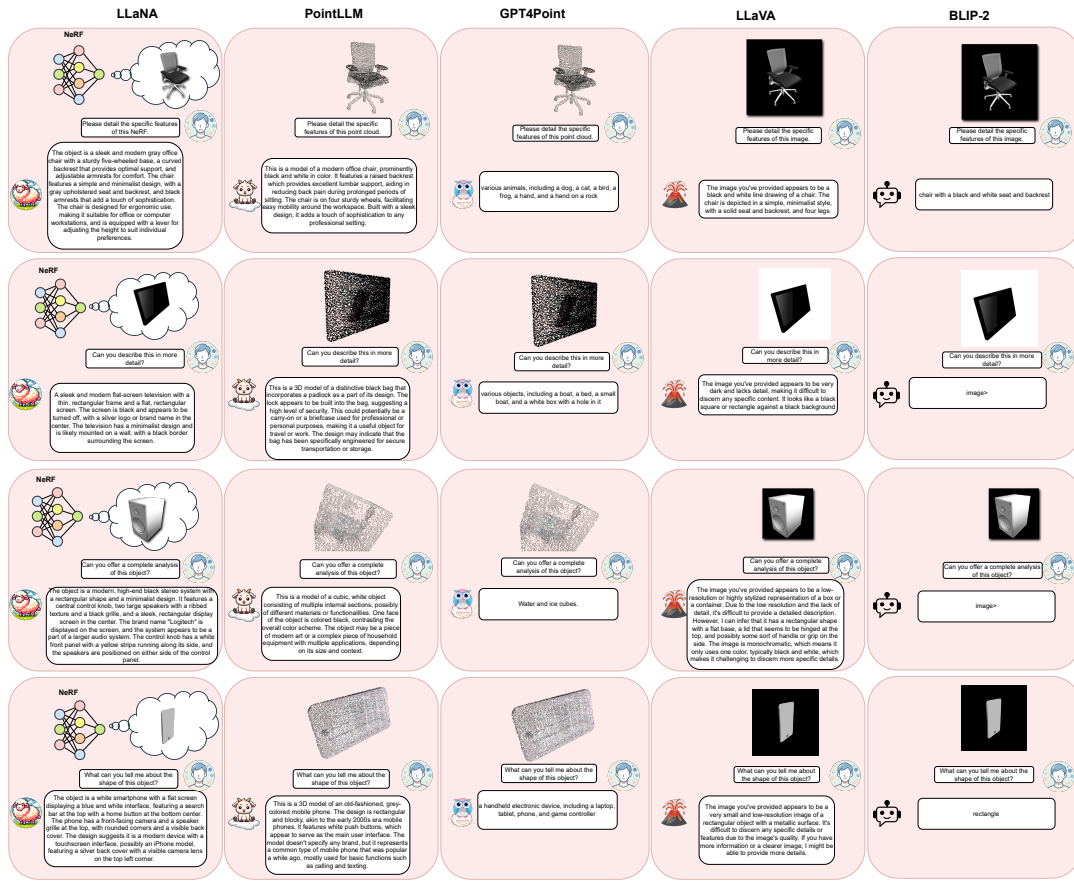


FIGURE 3.12: Detailed captioning example from ShapeNeRF-Text.



FIGURE 3.13: Single-round Q&A example from ShapeNeRF-Text.

3.6 Conclusion

In this Chapter, we have introduced *LLaNA*, the first Multimodal Large Language Model capable of directly ingesting the weights of Neural Radiance Fields. By utilizing a meta-network encoder to project NeRF weights into the embedding space of a pre-trained LLM, we have demonstrated that it is possible to perform complex reasoning tasks such as captioning, Q&A, and zero-shot classification without the need for intermediate renderings or discrete geometry extraction. Our large-scale NeRF-language benchmark, comprising over 320K samples from ShapeNet and Objaverse, has further validated that operating directly on neural weights is not only faster but also more effective than relying on resolution-dependent proxies like point clouds or 2D images.

The success of *LLaNA* underscores a fundamental shift in 3D-language understanding: the weights of a coordinate-based neural network serve as a compact and holistic representation of an object. However, the development of this model also revealed a significant architectural bottleneck. Because *LLaNA* relies on a meta-encoder that compresses the entire NeRF into a single global embedding, the LLM views the scene through a "bottleneck" that lacks explicit spatial structure. While this is sufficient for identifying an object or describing its general attributes, it falters when the task requires a granular understanding of the scene. Specifically, *LLaNA* struggles with spatial QA and scenes containing multiple objects, where the relationships between parts are as important as the whole.

This limitation suggests that while neural weights contain the necessary information, a single global token is insufficient to "unlock" the full spatial potential of a NeRF. For an assistant to truly navigate a complex scene or describe the relative positions of its components, it requires a representation that is not just holistic, but localized.

This realization provides the motivation for the next Chapter. Having proven that LLMs can "read" neural weights, we now seek to improve how they read them. We transition from a global weight-to-token paradigm to a spatially-aware architecture. In the following Chapter, we introduce *S-LLaNA*, a model designed to turn the information distributed across NeRF weights into localized, spatially-grounded tokens. By moving beyond a single global representation, we aim to bridge the gap between neural representations and the fine-grained spatial reasoning required for a truly versatile 3D assistant.

Chapter 4

Spatially-aware Weights Tokenization for NeRF-Language Models

This Chapter refers to the work “*Spatially-aware Weights Tokenization for NeRF-Language Models*” [7], published at the peer-reviewed Conference on Neural Information Processing Systems (NeurIPS) 2025.

4.1 Introduction

While the previous Chapter proved the feasibility of directly ingesting Neural Radiance Fields (NeRFs) into Multimodal Large Language Models (MLLMs), it primarily focused on capturing a holistic, global representation of the scene. As demonstrated with the introduction of *LLaNA* [4], processing the MLP weights of a NeRF allows for efficient captioning and Q&A without the computational overhead of rendering or the resolution limits of point clouds.

However, a significant challenge remains: **granularity**. While the meta-encoder strategy is remarkably faster than image-based alternatives, the reliance on a single global token to represent a whole NeRF inherently limits the ability of the model to capture fine-grained details and understand precise spatial relationships among objects, as illustrated in Figure 4.1 (left). In traditional MLLMs operating on images [74] or point clouds [133], spatial reasoning is typically achieved by dividing inputs into localized regions (patches or sets). NeRF weights, by contrast, lack an explicit spatial structure; the information is distributed across the MLP, with no single weight directly associated with a specific part of the object.

This observation leads to the central research question of this Chapter:
Is it possible to elicit a spatially-aware representation from the weights of a NeRF?

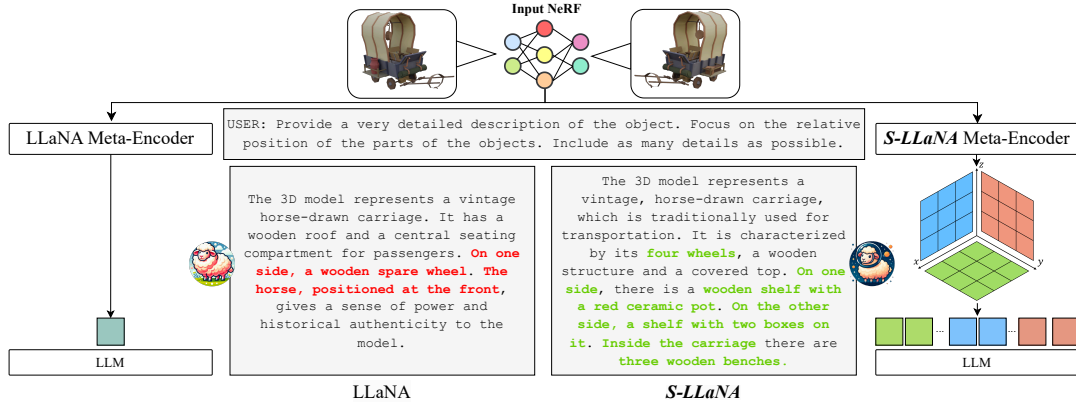


FIGURE 4.1: (left) LLaNA relies on a single global representation of the input NeRF. (right) *S-LLaNA* processes multiple spatially-aware tokens extracted by our new meta-encoder and, thus, can provide detailed descriptions that capture the relationships between objects. In particular, the answer of *S-LLaNA* contains more fine-grained information (e.g., it hints at the “*wooden shelf with a red ceramic pot*” and enumerates the “*four wheels*” and the “*three wooden benches*”) while also reasoning spatially (e.g., it discriminates between objects “*on one side*” and objects “*Inside the carriage*”).

To answer this, we propose a novel self-supervised framework, *weights2space*, trained to reorganize the information encoded within the weights of a NeRF into spatially-aware tokens. Specifically, *weights2space* employs a novel Transformer-based meta-encoder that ingests weights of a NeRF and outputs a sequence of tokens that are reorganized into a *tri-plane* [16] structure during training, so as to associate each token with a specific, localized spatial region. The meta-encoder is trained alongside a decoder to render images from the tri-plane representations. This approach allows the meta-encoder to extract a localized, spatially-grounded representation of the input NeRF. By leveraging this novel representation, we introduce *Spatial LLaNA* (*S-LLaNA*), a novel Multimodal Large Language Model for NeRFs which is capable of reasoning about fine-grained object details and spatial relationships, as shown in Figure 4.1, right.

To validate our approach, we thoroughly evaluate *S-LLaNA* on the existing NeRF-language datasets: ShapeNeRF [102], HST [5], and ObjNeRF [6]. Moreover, we create a novel dataset, *Spatial ObjNeRF*, consisting of 100 NeRF models from Objaverse [30] paired with detailed language descriptions focusing on object parts and their spatial relations. Experimental results demonstrate the substantial improvements offered by our approach in NeRF-language reasoning.

In summary, the research project presented in this Chapter proposes:

- *weights2space*: a self-supervised framework featuring a novel Transformer-based meta-encoder capable of extracting spatially-aware token representations directly from the weights of a NeRF.
- *Spatial LLaNA*: a novel MLLM for NeRFs that leverages the localized representations from the *weights2space* meta-encoder to achieve detailed spatial reasoning.
- *Spatial ObjNeRF*: a new manually annotated dataset of 100 NeRFs explicitly designed to evaluate detailed spatial reasoning tasks involving NeRFs.
- *Spatial LLaNA* achieves state-of-the-art performance on spatial reasoning tasks, as well as on existing benchmarks dealing with NeRF captioning and NeRF Q&A.

4.2 Related work

Learned representations in Multimodal Large Language Models. Multimodal Large Language Models (MLLMs) expand the capabilities of existing LLMs [33, 100, 122, 1] by combining language with other modalities like images, audio, or 3D data. Vision-language models [74, 63, 9] encode images using pre-trained Vision Transformers (ViT) [99] that convert image patches into sequences of visual tokens. These local representations capture fine-grained spatial information across the image. 3D MLLMs [133, 98, 139, 43] divide the input point cloud into local patches. Transformer-based encoders like PointBERT [136] and Recon++ [139] compute discrete token sequences that preserve spatial relationships. MLLMs for audio [44, 32, 141] split acoustic inputs into temporal patches and encode them using audio-specific encoders [19], producing sequences of tokens that retain temporal structure. Similarly, MLLMs for video-language understanding [64, 82, 142, 75, 81] use visual encoders to compute local features on each video frame, preserving both spatial and temporal information. UniPose [66] leverages a pose tokenizer to convert human pose data into discrete tokens that represent local body part configurations. In contrast to these local representation approaches, the only existing MLLM for Neural Radiance Fields (NeRFs), LLaNA [4], computes a single global embedding from the weights of a NeRF’s MLP. Although this approach enables basic language tasks on NeRFs, it critically overlooks spatial information since it does not adopt the local token-based representation strategy successfully used in other modalities. In this work, we propose *Spatial LLaNA*, a novel MLLM framework for NeRF which relies on our new *weights2space* meta-encoder to compute a set of spatial tokens from the weights of the input NeRF’s MLP,

preserving crucial spatial information while also allowing to extract richer and more detailed information from the input NeRF.

4.3 weights2space: spatially-aware NeRF weights tokenization

4.3.1 Architecture of weights2space

Our goal is to compute a spatially-aware representation of an object directly from the weights of its NeRF model. The only existing MLLM for NeRF, LLaNA [4], utilizes a global representation of the input NeRF, in the form of a single embedding. In contrast, state-of-the-art MLLMs typically operate on spatially-aware, token-based representations computed by Transformer encoders [99, 136, 139]. These architectures allow the model to capture localized features crucial for detailed understanding and downstream language tasks.

Inspired by these approaches, we introduce a novel self-supervised framework, *weights2space*, that learns how to transform the weights of a NeRF into a sequence of spatially-aware embeddings, as shown in Figure 4.2.

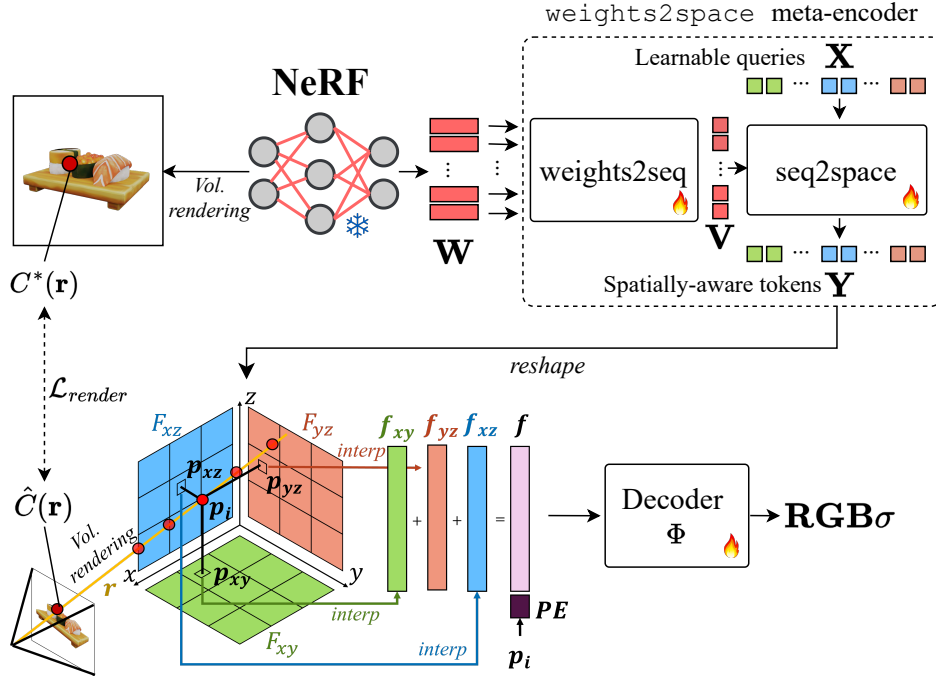


FIGURE 4.2: Overview of *weights2space*.

NeRF encoding. Similar to LLaNA [4], we assume NeRFs to be MLPs consisting of L hidden layers with H neurons, an input layer, and an output layer. The input layer has a weight matrix $\mathbf{W}_i \in \mathbb{R}^{I \times H}$ and biases $\mathbf{b}_i \in$

$\mathbb{R}^H$, where I depends on the frequency encoding [88] used to encode the input coordinates. Hidden layers have the same number of input and output neurons, H , thus their squared weight matrix has dimension $\mathbf{W}_l \in \mathbb{R}^{H \times H}$ and biases $\mathbf{b}_l \in \mathbb{R}^H$. The output layer, which predicts the *RGB* colour and density σ at a given 3D coordinate, has a weight matrix of $\mathbf{W}_o \in \mathbb{R}^{H \times 4}$ and biases $\mathbf{b}_{out} \in \mathbb{R}^4$. The parameters of this MLP are packed as in *nf2vec* [102] to provide the input to the meta-encoder. In particular, the weight matrices and biases of the input MLP are stacked along the row dimension to form a matrix $\mathbf{W} \in \mathbb{R}^{S \times H}$, where $S = I + L * (H + 1) + H + 2$. The weights and biases of the output layer, $\mathbf{W}_o$ and $\mathbf{b}_o$, are padded with zeros to obtain H columns. In our implementation, $L=3$, $H=64$, and $I=75$, leading to $S=336$.

Meta-encoder. Our meta-encoder processes the parameters of the input MLP packed into matrix $\mathbf{W}$ through two components: a *weights2seq* module and a *seq2space* module.

The *weights2seq* module, consisting of a series of linear layers – 4 in our experiments –, batch normalizations [48] and ReLU activations [3], processes each row of $\mathbf{W}$ independently for efficient computation. The module yields a set of tokens $\mathbf{V} \in \mathbb{R}^{S \times G}$, each representing a row of parameters in the input MLP, with $S = 336$ and $G = 1024$ in our setup. We note that this per-row processing module is the same as *nf2vec* [102], yet with one key modification: our architecture does not perform a final max-pooling to aggregate all row-level features into a single global representation. We conjecture that the pooled vector, which is the sole input utilized by LLaNA [4] to represent a NeRF, contains compact and coarse information on the underlying object. Instead, by skipping this pooling step, we aim at retaining more comprehensive and fine-grained information.

In standard MLLM pipelines for visual or 3D modalities, the encoder [99, 136] computes a sequence of spatially localized tokens, each representing a specific image patch or set of 3D points. However, the tokens computed by the *weights2seq* module have no direct correspondence to any specific spatial location or object part.

To elicit spatial information from the weights of the NeRF, we deploy the *seq2space* module, a Transformer decoder with 12 decoder layers, where the keys and values of the cross-attention layers are obtained from the sequence of tokens $\mathbf{V}$. Input to *seq2space* is a sequence of learnable queries $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N) \in \mathbb{R}^{N \times C}$, where N is the number of queries and C is the feature dimension. The Transformer decoder, hence, outputs a new sequence of tokens $\mathbf{Y} = (\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_N) \in \mathbb{R}^{N \times C}$. In our experiments, we employ

$N = 3072$ and $C = 516$.

Tri-plane representation and decoder. To force each token of $\mathbf{Y}$ to contain localized information, we reshape the N tokens of $\mathbf{Y}$ into a tri-plane feature map [16], defined as $M = (F_{xy}, F_{zx}, F_{yz})$. Each plane $F \in \mathbb{R}^{H \times W \times C}$, where H, W are the spatial dimensions, is obtained by arranging a subset of the tokens into a 2D grid. In particular, we reshape the 3072 tokens into three 32×32 spatial planes. Thus, each plane F has dimensions $H=W=32$ and $C=516$. In this way, each token of $\mathbf{Y}$ corresponds to a specific location in one of the three orthogonal planes. This design introduces spatial structure and locality into the otherwise unstructured token sequence originally obtained by the *weights2seq* module.

Given a 3D point $\mathbf{p} \in \mathbb{R}^3$, we can obtain its 3D feature by projecting it onto the three orthogonal planes to its corresponding 2D coordinates: $\mathbf{p}_{xy}, \mathbf{p}_{xz}, \mathbf{p}_{yz}$. Then, for each projection, bilinear interpolation is performed on the four nearest neighbors' features within the corresponding plane to extract three vectors of dimension C : $\mathbf{f}_{xy}, \mathbf{f}_{xz}, \mathbf{f}_{yz}$. These are summed element-wise to obtain a unified point-wise feature vector $\mathbf{f} \in \mathbb{R}^C$. A detailed explanation on this interpolation is reported in the publication this Chapter refers to [7].

This feature vector is then concatenated with a positional encoding [88] of $\mathbf{p}$, $\mathbf{PE}(\mathbf{p})$, and fed into an MLP decoder $\Phi(\cdot)$ that predicts the radiance field value $\hat{\mathbf{q}} = \Phi(\mathbf{p}, \theta)$, consisting of color and density components (R, G, B, σ) . Φ has the same architecture as the decoder used in [102]. It consists of 4 linear layers. A skip connection is applied after the second layer by adding a projection of the input. Each linear layer is followed by a ReLU activation, except for the final output layer, which is followed by a sigmoid for RGB and a shifted exponential for σ .

Training. All modules of *weights2space* are optimized end-to-end using differentiable volumetric rendering [65]. At each training step, a set of camera poses is sampled. For each pose, multiple rays are cast – together forming a training batch $\mathcal{R}$ – and 3D points are sampled along these rays. Each point is processed through the tri-plane module and the decoder as described above. The predicted radiance and density values are then aggregated along each ray using volumetric rendering to produce final RGB values. The framework is supervised using a smooth L1 loss between rendered and ground truth colors:

$$\mathcal{L}_{\text{render}} = \sum_{\mathbf{r} \in \mathcal{R}} \text{SmoothL1}(\hat{C}(\mathbf{r}) - C^*(\mathbf{r})) \quad (4.1)$$

where $C^*(\mathbf{r})$ is the ground truth RGB color for ray $\mathbf{r}$ and $\hat{C}(\mathbf{r})$ is the predicted

color by the decoder. Since NeRF can be a standalone modality with ground truth images unavailable, we adopt a self-supervised approach. Specifically, we render images directly from the input NeRFs using volumetric rendering. These pseudo ground truth images serve as reference images, making the entire training protocol fully self-supervised in the standalone NeRF scenario. The model is trained on 300K NeRFs from ShapeNeRF-Text and ObjNeRF-Text for 500 epochs with a learning rate of 0.00001, as done in LLaNA [6]. Training takes ~ 2 days on four 64GB A100 GPUs to reach convergence.

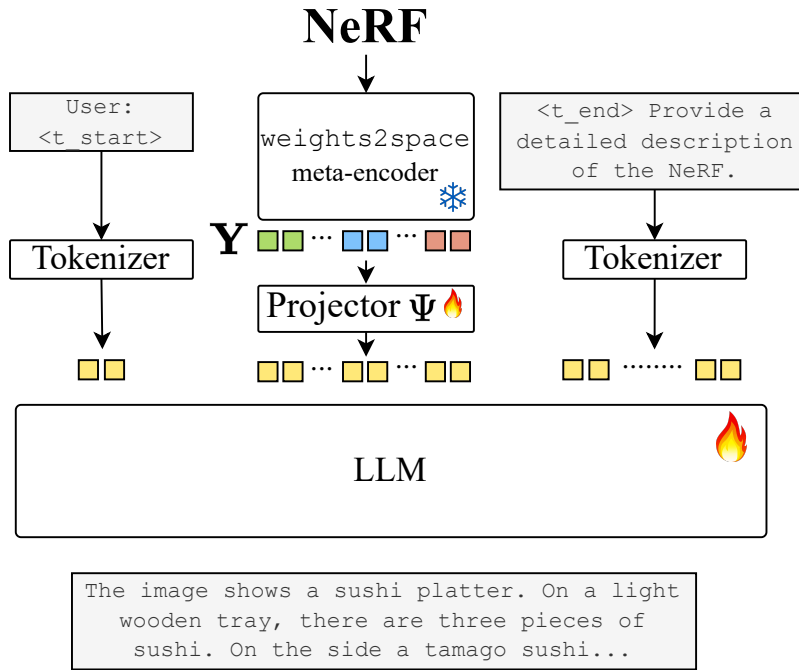


FIGURE 4.3: Overview of *Spatial LLaNA*.

4.3.2 Design Rationale: Weights-to-Tri-plane or Tri-plane-to-Language

A relevant question arises regarding the architecture of the input NeRF: if spatial locality is desired, why not restrict the framework to NeRFs that are natively parameterized as tri-planes or grids? While representations like Tri-planes [16] or Voxel grids [117] offer explicit spatial structures that are easier to tokenize, we deliberately chose to support the standard, coordinate-based MLP parameterization [88] for the following reasons:

Generalization. The MLP-based NeRF remains the fundamental baseline for neural representations. Many existing large-scale datasets and pre-trained models (e.g., those used in 3) are stored as MLP weights. By designing

weights2space to ingest these weights, we ensure our MLLM is compatible with the "standard" NeRF format without requiring the user to re-optimize every scene into a tri-plane format first.

Scientifically relevant contribution. From a research perspective, extracting spatiality from an unstructured set of weights is a more significant scientific contribution than simply subsampling a grid. We demonstrate that the information in an MLP is not "lost" or "flat": it is merely encoded within its weights. Our framework proves that a meta-encoder can learn to "de-serialize" these weights back into a spatially-meaningful 3D proxy.

Decoupling Input from Reasoning. By using the tri-plane only as an intermediate bottleneck during training, we gain the best of both worlds: we can ingest any compact MLP weight vector, but we reason using spatially-grounded tokens. This makes the system a flexible "adapter" rather than a rigid pipeline that only works with specific grid-based data structures.

4.4 *Spatial LLaNA*: spatially-aware MLLM for NeRF

Architecture. *Spatial LLaNA* (*S-LLaNA*) is the MLLM that performs NeRF-language tasks by taking as input the NeRF tokens $\mathbf{Y} \in \mathbb{R}^{N \times C}$ computed by the weights2space meta-encoder. This architecture is shown in Figure 4.3. Inspired by [4], we process these tokens with a shared projector network Ψ , composed of a stack of 2 trainable linear layers, interleaved with *GeLU* activation functions [41]. This network maps each NeRF token into the internal embedding space of the language model, LLaMA-2 [122]. The input to the LLM consists of the projected NeRF tokens, encapsulated between two learnable special tokens $\langle t_start \rangle$ and $\langle t_end \rangle$, and concatenated with a sequence of k language tokens corresponding to the user’s prompt. LLaMA processes the combined sequence to generate relevant textual outputs.

Training. We adopt the training protocol introduced in [6], using the ShapeNeRF-Text and ObjaNeRF-Text datasets. In a first stage, *S-LLaNA* is trained using brief textual descriptions, optimizing only the projection layers Ψ and the trainable special tokens $\langle t_start \rangle$ and $\langle t_end \rangle$. In a second stage, the full model, including the LLM, is fine-tuned using both brief and detailed descriptions, as well as Q&A annotations. Importantly, the encoder responsible for computing the NeRF tokens $\mathbf{Y}$ is kept frozen during both training stages. The total training set consists of approximately 300K NeRFs

with textual annotations. We supervise *S-LLaNA* by minimizing the negative log-likelihood of the text token at each position:

$$\mathcal{L}_{\text{CE}} = - \sum_{t=1}^T \log P(w_t \mid \hat{w}_0, \dots, \hat{w}_{t-2}, \hat{w}_{t-1}, \Psi(\mathbf{Y})) \quad (4.2)$$

where $P(w_t \mid \hat{w}_0, \dots, \hat{w}_{t-2}, \hat{w}_{t-1}, \Psi(\mathbf{Y}))$ is the probability assigned by the LLM to the correct token w_t at step t , conditioned on the previously predicted tokens and the projected sequence of NeRF tokens $\Psi(\mathbf{Y})$.

Implementation details. *S-LLaNA* shares the same LLM backbone as LLaNA [6], i.e., the 7B Vicuna and 13B Vicuna [25] of LLaMA-2 [122]. For experimental fairness, *S-LLaNA* follows the same training protocol as [6] and is trained on 300K samples from ShapeNeRF-Text and ObjaNeRF-Text for 3 epochs for each training stage. The projection network Ψ has 2 trainable linear layers which bring each token from dimension $C=516$ to the hidden dimension of the LLM, i.e., 4096 for LLaMA-7B and 5192 for LLaMa-13B. *S-LLaNA* is implemented in PyTorch and trained on NVIDIA A100 GPUs with 64GB of VRAM each. The 7B model requires 8 GPUs for training, while the 13B model requires 16 GPUs. Training either version of the model takes approximately one day.

The training configurations of each training stage are the following:

- **Stage 1:** Training on brief textual descriptions from ShapeNeRF-Text and ObjaNeRF-Text for 3 epochs, using a learning rate of 0.002. We use AdamW [78] as optimizer and a cosine learning rate scheduler.
- **Stage 2:** Training on brief and detailed textual descriptions, along with Q&A conversations from the same datasets, for 3 epochs, with a learning rate of 0.00002. We use AdamW [78] as optimizer and a cosine learning rate scheduler.

4.5 Experimental setup

Baselines. The only direct competitor of *S-LLaNA* is LLaNA [6], as this is the only MLLM performing NeRF-language tasks without rendering any 2D image or extracting an explicit 3D structure from input NeRFs. We also evaluate the same reference baselines as LLaNA [6], i.e. state-of-the-art MLLMs ingesting images or explicit 3D data like point clouds and meshes, with these models processing explicit data rendered or extracted from the input NeRF. As for 3D

MLLMs, included baselines are GPT4Point [98], PointLLM [133], 3D-LLM [43] and ShapeLLM [139]. As for 2D MLLMs, we consider LLaVA-1.6 [74] and BLIP-2 [63]. Since multiple views can be rendered from an input NeRF, these 2D baselines are evaluated using different viewpoints, i.e. front-view (FV), back-view (BV), random-view (RV), and in a multi-view (MV) scenario where we concatenate image tokens from $N = 3$ images rendered from random viewpoints. For each method, we report standard metrics employed in MLLM literature [6, 98, 133]: Sentence-BERT [103] and SimCSE [37] based on pre-trained encoders; BLEU-1 [93], ROUGE-L [71], and METEOR [11] based on n-gram statistics.

NeRF-language datasets. As in [6], we evaluate the baselines on the test sets of the existing NeRF-language datasets ShapeNeRF-Text and ObjaNeRF-Text. Two different test sets for ObjaNeRF-Text are considered to ensure fair comparison with PointLLM and GPT4Point. Results on these datasets are divided into three tasks: brief captioning, detailed captioning, and single-round Q&A conversation. In addition, within the brief captioning task, we evaluate the performance of the methods on HST [5], a subset of ShapeNet objects with manually annotated short captions.

Spatial ObjaNeRF. To evaluate the ability of NeRF-language models to capture spatial relationships and fine-grained geometric details, we introduce Spatial ObjaNeRF, a curated benchmark specifically designed for this purpose. This benchmark consists of 100 human-annotated 3D models from ObjaNeRF-Text. To test the spatial reasoning capability of MLLMs, each data sample is a scene featuring multiple objects. For each scene, we created a detailed textual description that emphasizes spatial structure, highlighting the size, shape, and relative positioning of specific objects. These annotations are designed to evaluate whether the MLLMs can reason about detailed geometry and fine-grained spatial layout. The MLLM models must extract and attend to localized features across the 3D structure to succeed. Examples of such annotations and their comparison with those present in ObjaNeRF-Text are provided in Figure 4.4.

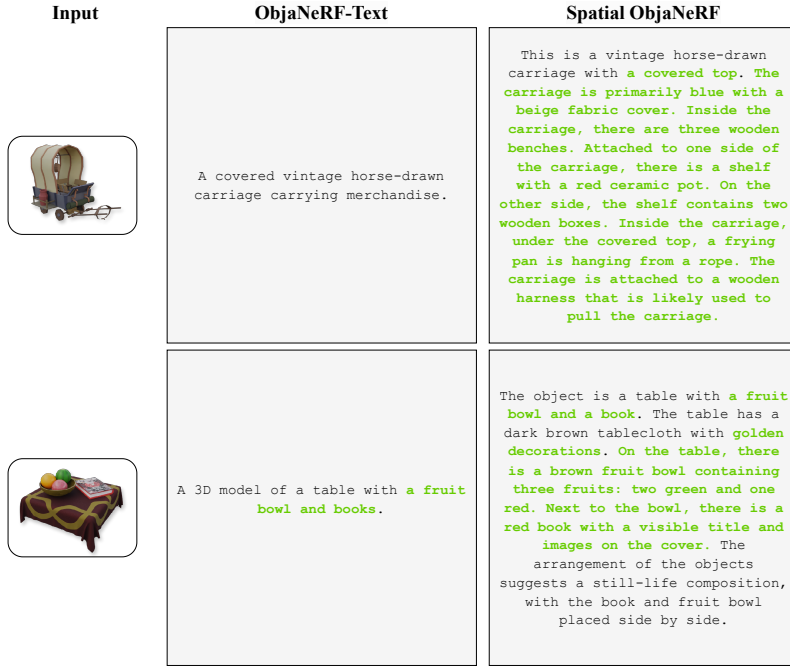


FIGURE 4.4: Comparison between the annotations of ObjNeRF-Text (GPT4Point test set) and Spatial ObjNeRF. In **green**, the fine-grained spatial details.

Model	Modality	ShapeNeRF-Text					HST				
		S-BERT	SimCSE	BLEU-1	ROUGE-L	METEOR	S-BERT	SimCSE	BLEU-1	ROUGE-L	METEOR
LLaVA-vicuna-7b	Image (FV)	59.85	62.35	22.67	23.24	23.35	54.31	56.28	10.08	14.71	14.53
LLaVA-vicuna-7b	Image (BV)	55.68	58.46	21.97	22.46	22.50	51.75	52.29	8.13	13.96	14.18
LLaVA-vicuna-7b	Image (MV)	59.77	61.42	23.16	22.68	23.01	54.15	56.87	10.32	12.76	15.13
BLIP-2 FlanT5-xxl	Image (FV)	56.13	58.21	5.46	18.69	9.67	57.11	59.43	8.21	18.02	12.14
BLIP-2 FlanT5-xxl	Image (BV)	52.48	54.05	5.67	18.20	9.50	54.11	56.37	9.09	17.38	11.79
LLaVA-vicuna-13b	Image (FV)	61.00	61.16	14.30	20.00	23.31	55.62	55.56	6.56	11.81	14.52
LLaVA-vicuna-13b	Image (BV)	54.35	56.09	21.94	21.67	22.09	50.00	50.79	9.39	12.76	14.46
LLaVA-vicuna-13b	Image (MV)	59.64	61.01	22.84	22.17	23.08	54.25	55.56	9.78	14.13	14.99
3D-LLM FlanT5-xl	Mesh + MV	59.46	56.42	12.69	21.49	14.32	56.07	52.13	15.94	20.71	15.22
GPT4Point-Opt-2.7b	Point cloud	41.85	40.22	11.76	16.54	11.63	43.15	42.22	12.02	18.73	13.69
PointLLM-7b	Point cloud	49.59	48.84	16.74	17.92	14.56	43.40	44.50	8.53	11.64	9.97
ShapeLLM-7b	Point cloud	39.16	38.34	20.71	21.35	17.50	32.72	35.66	9.39	11.33	9.76
PointLLM-13b	Point cloud	51.54	50.35	17.20	18.51	14.92	45.41	46.39	9.57	12.38	11.92
ShapeLLM-13b	Point cloud	42.29	40.18	21.59	21.87	17.96	25.09	38.24	10.03	13.26	9.94
LLaNA-7b	NeRF	74.94	76.41	42.73	43.64	41.95	64.78	65.01	20.65	23.24	24.10
S-LLaNA-7b	NeRF	78.91	79.91	47.81	49.08	47.58	67.76	68.63	20.31	23.61	24.47
LLaNA-13b	NeRF	75.09	76.45	42.83	43.70	42.21	65.66	66.09	20.80	24.28	25.90
S-LLaNA-13b	NeRF	78.98	79.98	47.97	49.22	47.87	67.43	68.26	20.91	23.86	25.03

TABLE 4.1: NeRF brief captioning on ShapeNeRF-Text and HST datasets. Best results are in **bold**, runner-up is underlined. (FV: front-view, BV: back-view, MV: multi-view)

4.6 Experimental results

4.6.1 Results on ShapeNeRF-Text, HST, and ObjNeRF-Text.

We evaluate *S-LLaNA* on the tasks of NeRF brief captioning, detailed captioning, and Q&A on ShapeNeRF-Text, HST, and ObjNeRF-Text. Results are presented in Tables 4.1, 4.2, 4.3, and 4.4. Across all benchmarks and tasks, *S-LLaNA* consistently outperforms LLaNA and all other 2D and 3D

Model	Modality	ObjaNeRF-Text (PointLLM test set)					ObjaNeRF-Text (GPT4Point test set)				
		S-BERT	SimCSE	BLEU-1	ROUGE-L	METEOR	S-BERT	SimCSE	BLEU-1	ROUGE-L	METEOR
LLaVA-vicuna-7b	Image (RV)	39.29	40.93	5.22	8.35	12.49	41.59	42.38	7.24	11.74	16.01
LLaVA-vicuna-7b	Image (MV)	40.15	41.09	5.62	9.07	13.22	42.34	43.21	8.59	11.94	18.52
BLIP-2 FlanT5-xxl	Image (RV)	35.48	35.19	7.69	13.75	11.68	37.24	37.32	10.63	16.87	14.69
LLaVA-vicuna-13b	Image (RV)	38.57	39.05	4.62	7.63	11.85	42.08	41.04	6.53	10.57	15.67
LLaVA-vicuna-13b	Image (MV)	41.01	41.10	4.87	8.03	12.35	44.15	43.19	6.75	10.86	16.10
3D-LLM FlanT5-xl	Mesh + MV	38.22	39.60	5.47	7.29	10.75	41.26	37.74	11.08	12.26	15.58
GPT4Point-Opt-2.7b	Point cloud	-	-	-	-	-	34.42	31.89	8.76	13.62	13.55
PointLLM-7b	Point cloud	38.81	40.11	5.55	8.39	11.24	-	-	-	-	-
ShapeLLM-7b	Point cloud	30.13	32.42	5.80	7.72	11.38	25.92	25.57	8.04	9.60	13.23
PointLLM-13b	Point cloud	39.64	40.63	5.94	8.43	11.63	-	-	-	-	-
ShapeLLM-13b	Point cloud	31.44	32.69	6.01	7.59	11.66	26.87	27.03	8.31	9.65	14.10
LLaNA-7b	NeRF	41.36	42.28	13.21	16.93	15.63	43.73	43.09	20.22	25.15	22.21
<i>S-LLaNA-7b</i>	NeRF	<u>45.23</u>	<u>45.80</u>	<u>15.72</u>	<u>19.98</u>	<u>17.70</u>	<u>50.12</u>	<u>49.55</u>	<u>24.49</u>	<u>30.72</u>	<u>26.15</u>
LLaNA-13b	NeRF	42.08	42.40	13.86	17.51	16.18	44.26	43.75	20.61	25.62	22.36
<i>S-LLaNA-13b</i>	NeRF	45.44	46.18	15.78	20.17	<u>17.61</u>	50.68	50.30	24.53	<u>30.52</u>	26.41

TABLE 4.2: NeRF brief captioning on ObjaNeRF-Text. Best results are in **bold**, runner-up is underlined. (RV: random view, MV: multi-view)

Model	Modality	S-BERT	SimCSE	BLEU-1	ROUGE-L	METEOR
LLaVA-vicuna-7b	Image (FV)	57.55	57.68	14.99	22.82	14.36
LLaVA-vicuna-7b	Image (BV)	53.11	54.46	14.73	22.47	14.05
LLaVA-vicuna-7b	Image (MV)	55.26	58.46	15.07	24.05	14.85
BLIP-2 FlanT5-xxl	Image (FV)	41.27	40.69	0.18	7.83	2.60
BLIP-2 FlanT5-xxl	Image (BV)	38.49	37.89	0.19	7.72	2.58
LLaVA-vicuna-13b	Image (FV)	59.08	58.87	23.63	23.55	22.55
LLaVA-vicuna-13b	Image (BV)	50.09	50.33	13.77	21.36	13.18
LLaVA-vicuna-13b	Image (MV)	60.21	59.51	15.07	32.16	14.64
3D-LLM FlanT5-xl	Mesh + MV	58.00	53.91	1.58	14.40	5.28
GPT4Point-Opt-2.7b	Point cloud	42.44	38.33	3.72	9.21	5.13
PointLLM-7b	Point cloud	59.02	58.30	10.28	19.26	10.55
ShapeLLM-7b	Point cloud	43.45	40.92	8.18	18.47	10.15
PointLLM-13b	Point cloud	59.64	58.55	10.52	19.44	10.83
ShapeLLM-13b	Point cloud	43.26	40.98	10.47	18.42	10.20
LLaNA-7b	NeRF	75.25	77.42	19.57	32.96	20.45
<i>S-LLaNA-7b</i>	NeRF	81.03	83.00	<u>44.74</u>	<u>40.08</u>	<u>36.03</u>
LLaNA-13b	NeRF	75.51	77.63	19.87	32.93	20.46
<i>S-LLaNA-13b</i>	NeRF	<u>80.98</u>	<u>82.90</u>	44.78	40.09	36.24

TABLE 4.3: NeRF detailed captioning on ShapeNeRF-Text.

Model	Modality	S-BERT	SimCSE	BLEU-1	ROUGE-L	METEOR
LLaVA-vicuna-7b	Image (FV)	71.79	71.96	25.79	34.04	34.86
LLaVA-vicuna-7b	Image (BV)	70.88	70.93	25.17	33.30	34.22
LLaVA-vicuna-7b	Image (MV)	72.26	70.67	24.09	34.67	35.64
BLIP-2 FlanT5-xxl	Image (FV)	45.20	47.92	11.50	20.16	13.49
BLIP-2 FlanT5-xxl	Image (BV)	45.06	47.66	11.50	19.98	13.44
LLaVA-vicuna-13b	Image (FV)	71.61	70.98	20.19	30.42	32.53
LLaVA-vicuna-13b	Image (BV)	68.25	69.06	20.03	29.84	32.27
LLaVA-vicuna-13b	Image (MV)	71.84	71.16	20.04	30.20	33.46
3D-LLM FlanT5-xl	Mesh + MV	69.62	67.55	32.19	40.95	35.83
GPT4Point-Opt-2.7b	Point cloud	27.62	31.41	6.26	9.38	5.41
PointLLM-7b	Point cloud	74.70	74.40	36.81	44.41	39.76
ShapeLLM-7b	Point cloud	46.60	46.32	19.47	17.13	12.83
PointLLM-13b	Point cloud	74.65	74.18	37.16	44.86	40.13
ShapeLLM-13b	Point cloud	47.22	46.59	20.03	17.45	13.22
LLaNA-7b	NeRF	81.03	81.61	45.98	53.27	49.97
<i>S-LLaNA-7b</i>	NeRF	<u>83.49</u>	<u>84.06</u>	50.89	58.39	<u>54.52</u>
LLaNA-13b	NeRF	81.05	81.60	46.08	53.44	49.98
<i>S-LLaNA-13b</i>	NeRF	83.60	84.17	<u>50.86</u>	<u>58.25</u>	54.58

TABLE 4.4: NeRF single-round Q&A on ShapeNeRF-Text.

MLLM baselines, often by substantial margins. The 13B variant of *S-LLaNA*

USER: What kind of object is illustrated by this NeRF?

Input	<i>S-LLaNA</i>	LLaNA	PointLLM	LLaVA	ground-truth
	The object is a flat-screen TV with a black frame , gray screen , and a stand with a dark grey base and silver drawers .	A grey, rectangular box with a flat top , a slightly recessed front panel, and two visible screws on the top left and right corners .	The 3D object model showcases a rectangular vertical slab , colored in shades of silver and black . Its composition and design suggest a minimalist and chic aesthetic.	The image appears to be a 3D model or a digital rendering of a television stand or entertainment center. It features a flat-screen TV mounted on a stand with a dark finish , possibly made of wood or a similar material. The TV is turned off, as indicated by the blank screen. The background is black, which highlights the stand and TV, making them the focal point of the image.	A flat-screen TV mounted on a dark-colored stand with a shelf below , featuring a sleek and modern design.
	A 3D model of a white toy lion with a golden mane and tail .	Low poly 3D model of a beagle dog wearing a vest .	The 3D model represents a charming plush puppy teddy bear , which is designed to resemble a playful canine. The toy features a snowy white material, making it appear soft and cuddly.	The image you've provided appears to be a 3D representation of an animal, possibly a sheep or a similar woolly creature .	A ceramic lion with a golden mane and a golden tail .
	A 3D model of a yellow and black motorcycle tire with a yellow rim .	A 3D model of a yellow and black ring-shaped spaceship with two wheels and a blue ball .	The object in the point cloud is a car wheel with a tyre . It appears to be a standard-sized wheel found on many types of passenger vehicles. The tyre appears to be made of rubber and properly inflated, indicating it's ready for use.	The image you've provided appears to be a point cloud representation of a black object , which is a common way to visualize 3D data from a LiDAR sensor.	A black tire with a yellow midpiece .

FIGURE 4.5: Qualitative evaluation of the brief captioning task on ShapeNeRF-Text and ObjaNeRF-Text. In **red**, the hallucinations generated by the MLLMs. In **green**, the correct fine-grained and spatial details.

achieves the highest performance in nearly every setting, while the 7B variant is typically the second-best model, outperforming all the other baselines. These results demonstrate the effectiveness of the NeRF spatially-aware tokenized representation learned by our weights2space meta-encoder in existing public NeRF-language datasets. On the brief captioning task, reported in Tables 4.1 and 4.2, *S-LLaNA* outperforms LLaNA by approximately ~ 4 points on Sentence-BERT and SimCSE metrics, consistently across all datasets and model sizes. This indicates that even in tasks requiring high-level object descriptions, the spatially-aware NeRF tokens produced by the weights2space meta-encoder provide more discriminative and semantically rich input than the global representation of LLaNA. Table 4.4 shows that, on the Q&A task, *S-LLaNA* outperforms LLaNA with performance improvements comparable to those observed in brief captioning. This suggests that the ability to reason over object structure and part-level features, supported by our spatial encoding, is also beneficial to this task. We observe larger gains on the detailed captioning task, in Table 4.3, where *S-LLaNA* surpasses LLaNA by up to 6 points on Sentence-BERT and SimCSE and over 10 points on reference-based metrics. These results confirm that the rich, fine-grained geometric information captured by the weights2space meta-encoder helps guide the language model toward more complete and structured descriptions. All the MLLMs operating on explicit data structures, i.e., Vision Language Models and 3D MLLMs, show significantly worse performance than LLaNA and *S-LLaNA*, highlighting the effectiveness of building NeRF-language models operating directly on NeRF weights. Figures 4.5, 4.6, and 4.7 show qualitative examples of *S-LLaNA* against the baselines on these NeRF-language tasks.



FIGURE 4.6: Qualitative evaluation of the detailed captioning task on ShapeNeRF-Text. In **red**, the hallucinations generated by the MLLMs. In **green**, the correct fine-grained and spatial details.

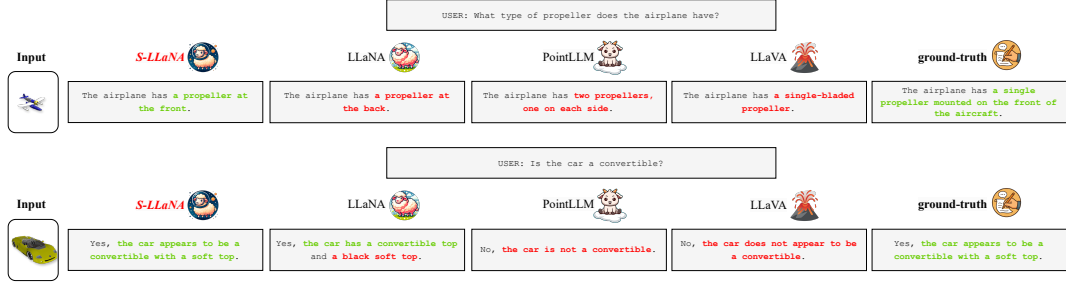


FIGURE 4.7: Qualitative evaluation of the single-round Q&A task on ShapeNeRF-Text. In **red**, the hallucinations generated by the MLLMs. In **green**, the correct fine-grained and spatial details.

4.6.2 Results on Spatial ObjNeRF

Model	Modality	S-BERT	SimCSE	BLEU-1	ROUGE-L	METEOR
LLaVA-vicuna-7b	Image (RV)	70.11	73.25	21.95	24.26	20.09
LLaVA-vicuna-7b	Image (MV)	72.13	74.98	22.44	26.87	21.33
BLIP-2 FlanT5-xxl	Image (RV)	54.12	60.23	8.47	10.23	6.59
LLaVA-vicuna-13b	Image (RV)	70.55	75.82	21.31	29.45	19.74
LLaVA-vicuna-13b	Image (MV)	73.42	77.06	24.20	<u>30.85</u>	20.47
3D-LLM FlanT5-xl	Mesh + MV	53.24	54.69	12.05	14.57	16.88
GPT4Point-Opt-2.7b	Point cloud	40.08	40.72	4.65	11.71	5.88
PointLLM-7b	Point cloud	72.67	73.93	25.90	24.75	19.05
ShapeLLM-7b	Point cloud	62.48	64.64	9.26	21.55	17.69
PointLLM-13b	Point cloud	73.08	75.07	26.39	25.02	19.18
ShapeLLM-13b	Point cloud	64.22	67.98	9.55	21.07	16.84
LLaNA-7b	NeRF	70.61	71.77	25.61	25.66	19.21
S-LLaNA-7b	NeRF	<u>75.01</u>	<u>77.30</u>	<u>29.43</u>	28.85	<u>21.70</u>
LLaNA-13b	NeRF	73.20	74.76	26.78	29.15	20.37
S-LLaNA-13b	NeRF	78.25	80.58	32.34	31.03	24.56

TABLE 4.5: Detailed spatial captioning on Spatial ObjNeRF. Best results are in **bold**, runner-up is underlined. (RV: random view, MV: multi-view)

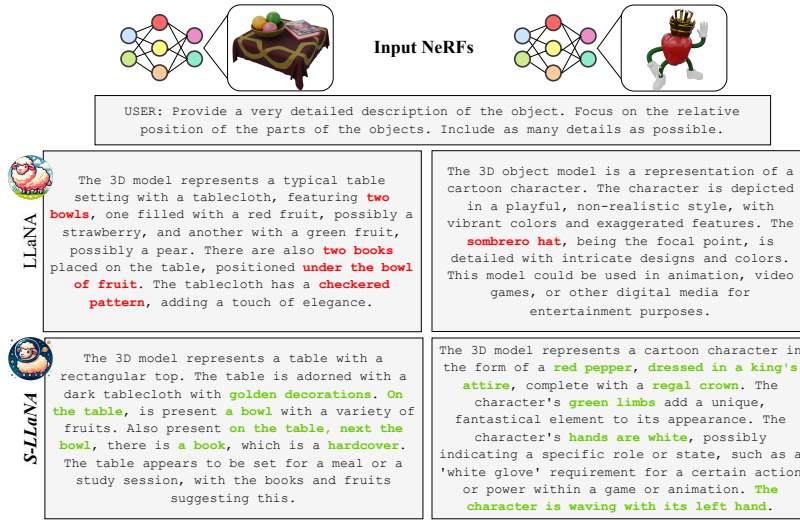


FIGURE 4.8: Qualitative examples on Spatial ObjaNeRF

Unlike previous captioning tasks, Spatial ObjaNeRF targets explicitly the understanding of relative positions, part-whole hierarchies, and size comparisons: skills that require a structured and localized representation of geometry. This benchmark, described in Section 4.5, is proposed as a key tool to assess whether NeRF-language models can achieve fine-grained 3D understanding. Quantitative results on Spatial ObjaNeRF are reported in Table 4.5. *S-LLaNA-13b* achieves the best performance across all metrics, followed by *S-LLaNA-7b*. The performance gap between *S-LLaNA* and *LLaNA* highlights the importance of explicitly modeling spatial features within the representation provided as input to the LLM. While *LLaNA* relies on a global embedding computed by processing the weights of the NeRF’s MLP with linear layers, *S-LLaNA* benefits from spatially-aware tokens extracted by the `weights2space` meta-encoder. Qualitative examples in Figure 4.8 illustrate this advantage. In the first example, only *S-LLaNA* accurately recognizes the objects on the table and correctly describes their spatial arrangement. Indeed, it notes that the book is “next to the bowl”. *LLaNA* mistakenly reports the presence of “two bowls” and “two books” and describes their relative position by stating that the books are “under the bowl of fruit”. In the second example, *S-LLaNA* demonstrates precise part-level reasoning, describing the hat of the character as a “regal crown”, while *LLaNA* incorrectly describes it as a “sombrero”. These examples further demonstrate the superior ability of *S-LLaNA* to interpret complex scenes with fine-grained details, particularly when multiple objects interact spatially. More qualitative examples on Spatial ObjaNeRF are shown in Figure 4.9.

USER: Provide a detailed description of the object. Focus on the relative position of the parts of the object. Include as many details as possible.					
Input	S-LLaNA	LLaNA	PointLLM	LLaVA	ground-truth
	The 3D model represents a table with a rectangular top. The table is adorned with a dark tablecloth with golden decorations. On the table, there is a bowl of fruit. Next to the bowl, there is a book, which is a hardcover. The table appears to be set for a meal or a study session, with the bowl of fruit and a book placed on the table suggesting this.	The 3D model represents a typical table setting with a tablecloth, featuring two bowls, one filled with a red fruit, possibly a strawberry, and another with a green fruit, possibly a pear. There are also two books placed on the table, positioned under the bowls of fruit. The tablecloth has a checkered pattern, adding a touch of elegance to the overall appearance.	A 3D model of a compact, square-shaped table with a flat, stable top surface. On the table is a small circular white lamp, with a futuristic and interesting design. Positioned on the table near the fruit bowl are two hardcover black books. The objects collectively suggest a setting of leisure reading while consuming fresh fruits.	This is the 3D model of a table with a fruit bowl and books. The fruit bowl is centrally placed on the table, containing green, yellow, and red fruits. Next to the bowl, there are two books. This setting represents a layout suitable for long study sessions.	The object is a table with a fruit bowl and a book. The table has a dark brown tablecloth with golden decorations. On the table, there is a brown fruit bowl containing three fruits: two green and one red. Next to the bowl, there is a red book with a visible title and images on the cover. The arrangement of the objects suggests a still-life composition, with the book and fruit bowl placed side by side.
	The 3D model depicts an old, neglected speaker placed beside a wooden crate. The speaker, appearing worn out, is likely non-functional at present. The crate, seeming sturdy and durable, could potentially be used for storage or transportation of goods. In this setting, the size of the speaker is much larger than the size of the wooden crate. This setting suggests a scene of abandonment, possibly in a post-apocalyptic scenario or a rustic, vintage-style environment.	The 3D model represents an old, weathered speaker that appears to have been discarded or left behind. It is lying in a dilapidated state, showing signs of rust and decay. On top of the speaker is a wooden crate, perhaps used for storage or transportation.	The 3D model represents two distinct items: a cuboid grey box and a rectangular wooden crate. The box, showcasing a mixture of white and grey colors, seems to be a standard contemporary design. The wooden crate, presented in a natural, brown tone, appears to be sizable and sturdy, implying its use for either storage or transportation of goods.	This is a large, dark-colored speaker with a visible front grille and a label or brand marking on the top. The speaker is primarily black with a grey frame. It can be used for amplification at great music shows. The background is not visible, but it is probably a large area for shows.	This is a 3D model of a large black speaker and a brown wooden crate. The black speaker is placed next to the wooden crate. The speaker has a large rectangular body with a large circular component in the center and a smaller one at the top right angle. The crate has a wooden texture with a visible grain pattern and a natural brown color. The speaker is much larger than the wooden crate.
	The 3D object model is a pixelated, green-colored spider with a cubic body, similar to the popular game Minecraft. The spider features a cubic body, with green borders accentuating its edges. It has four legs, extending from the corners of the body. The spider's body is divided into multiple cubes, giving it a unique, pixelated aesthetic. This model could be used in Minecraft-themed animations, fan creations, or graphics.	The 3D model represents a robotic spider with a unique design. Its body is shaped like a cube, with vibrant green coloring. The spider has discernible eyes and mouth, and its legs are distinguishable. The spider has eight legs and prominent eyes. This spider model may be used in various applications such as video games, animations, or robotics research and development.	The 3D object model represents a stylized, cartoonish representation of a spider. Its main body is shaped like a cube, further differentiating it from a typical spider model. The coloring is mostly green, including the borders around the object and the legs. Despite its unusual form, it maintains the standard number of legs typically associated with spiders. It could be imagined that this object serves as a decorative item or a kid's toy, thanks to its vibrant coloring and simplified, friendly design.	The object is a green pixelated spider with a cubic body, green borders, and legs. The body is composed of green pixels, and the spider has eight legs that are evenly spaced. This model may be used in animation or video game design.	The object is a pixelated representation of a spider with a cubic body. The body is composed of green and yellow pixels, and the spider has four legs extended from the corners of its cubic body. The spider's legs are positioned in a typical spider arrangement, with two pairs on the front and two at the back. The spider's body is square-shaped. The head of the spider is not represented. There is a small square protruding from the body of the spider.
	The 3D object model depicts a whimsical scene featuring a purple octopus crawling out from a traditional red telephone booth. The booth is situated against a brick wall, which provides a historical and nostalgic backdrop. Next to the octopus, there is a grey street lamp. This model could be used in animation, game design, or as a visual element in storytelling to evoke a sense of wonder and familiarity.	The object is a unique and intriguing scene featuring a vibrant purple octopus, which appears to be made of a smooth and sleek material, positioned in front of a classic red telephone booth. The background is a simple brick wall, providing a neutral and unobtrusive setting for the scene.	A striking and unusual scene is presented in the 3D model. A vibrant purple octopus is depicted in the foreground, while a classic red telephone booth stands beside it. The octopus, with its iconic tentacles, occupies most of the space in the picture, while the telephone booth, though retro in style, adds a touch of nostalgia. The brick wall behind them forms a stark background, contributing to the overall aesthetic of the image. This object could be part of an urban or pop culture scene, perhaps intended for artistic or illustrative purposes, or even as a peculiar decorative piece.	The image depicts a red telephone booth in front of a red brick wall. The octopus is positioned to the left of the telephone booth, which is set against a brick wall. The design is typical of a 3D video game.	The object is a 3D scene with a red telephone booth, in contact with a red brick wall and a purple octopus next to the telephone booth. The octopus has a textured body with a pattern of purple and white, and its tentacles are spread out, giving the impression of it crawling out from the telephone booth. The telephone booth is a classic red color with a red roof and the word "TELEPHONE" written in black on a white background. The brick wall is a reddish-brown color with a textured surface, and the floor appears to be a grayish color with a pattern that could be interpreted as tiles. The octopus is hanging from the lamp post towards the ground and the telephone booth.
	The 3D object model represents a marine organism, specifically a starfish, which is presented in a lifelike pink color. The starfish is characterized by two distinctly dissected legs, indicating that it is in a static pose. One of them is truncated, while the other reveals the internal structure of the starfish. The model is mounted on a wooden wall, suggesting it might be a part of a marine biology study, a teaching aid in a classroom, or a decorative piece.	The 3D object model represents a starfish, distinctly colored in a shade of pink. It has two arms, one of which is presented in a dissected state, revealing the internal structure of the starfish. The model is mounted on a wooden wall, suggesting it might be a part of a marine biology study, a teaching aid in a classroom, or a decorative piece.	A detailed 3D model of a pink starfish specimen, carefully mounted on a wooden wall. The starfish is in a natural pose, with its five pointed arms extended as usual. Two of these arms have been carefully dissected, lying separate from the rest of the body. This object serves an educational purpose, useful in biology classes for demonstrating marine life anatomy, or in museums and exhibits for public display and learning.	The starfish specimen is pink with a central yellow and blue band. It has two dissected legs, one on each side, which are also pink. The specimen is mounted on a wooden wall, which is a light brown color and makes a good contrast with the vivid pink color of the starfish. This object can be used as a decorative piece of art.	The object is the 3D model of a pink starfish attached to a wooden panel. The top left leg is dissected, while the top right leg has yellow and red colors and it shows the internal part of the starfish. The center of the starfish has blue and yellow color. The wooden panel, lastly, has a slight backward tilt.

FIGURE 4.9: Qualitative evaluation on Spatial ObjaNeRF. In red, the hallucinations generated by the MLLMs. In green, the correct fine-grained and spatial details.

4.6.3 Failure cases of Spatial LLaNA

Figure 4.10 presents some failure cases of *Spatial LLaNA* on Spatial ObjaNeRF. These failures can be broadly categorized into two main categories: (i) the model fails to recognize the main object in the scene (first and second row), or (ii) the model generates short descriptions lacking specific details (third row). Such failures may happen in cases where out-of-distribution objects are provided as input to the model. For example, in the first row, the model is fooled by the unusual shape of the teapot, leading to the weird description “a square device for making tea”, while the “pink teapot” had already been correctly included in the answer. In the second row, since most of the snowmen of ObjaNeRF-Text are simple white snowballs with outstretched



FIGURE 4.10: Examples of failure cases of *Spatial LLaNA* on Spatial ObjNeRF. In **red**, the hallucinations generated by the model. In **green**, the correct fine-grained and spatial details.

Model	LLM Finetuned	# Trainable params	SN-T brief captioning		ON-T brief captioning		SN-T detailed captioning		SN-T Q&A		Spatial test	
			S-BERT	SimCSE	S-BERT	SimCSE	S-BERT	SimCSE	S-BERT	SimCSE	S-BERT	SimCSE
LLaNA-7b	✓	6.75B	74.94	76.41	41.36	42.28	75.25	77.42	81.03	81.61	70.61	71.77
LLaNA-7b		143M	67.75	68.88	33.26	33.73	71.25	71.74	71.25	71.74	62.48	61.44
S-LLaNA-7b	✓	6.75B	78.91	79.91	45.23	45.80	81.03	83.00	83.49	84.06	73.20	74.76
S-LLaNA-7b		140M	79.29	80.36	45.17	45.81	81.03	82.90	82.43	83.01	73.27	70.99
LLaNA-13b	✓	13.03B	75.09	76.45	42.08	42.40	75.51	77.63	81.05	81.60	75.01	77.30
LLaNA-13b		178M	70.48	68.85	35.21	36.68	72.66	70.41	72.24	72.62	62.34	61.50
S-LLaNA-13b	✓	13.03B	78.98	79.98	45.44	46.18	80.98	82.90	83.60	84.17	78.25	80.58
S-LLaNA-13b		175M	79.05	80.17	44.87	45.41	80.63	82.50	82.10	82.67	77.72	80.47

TABLE 4.6: Experiments on LLM finetuning. Best results are in **bold**, runner-up is underlined. (SN-T: ShapeNeRF-Text, ON-T: ObjNeRF-Text PointLLM test set)

arms made of sticks, this particular snowman is incorrectly described as a “cartoon character”. Another interesting case appears in the last row, where *Spatial LLaNA* correctly detects a “blue toy truck” but omits any distinctive features in the generated description. This may be attributed to the complexity or uniqueness of the object, which challenges the ability of *Spatial LLaNA* to generate a detailed description.

4.6.4 Ablation on LLM finetuning

As described in Section 4.5, both LLaNA and *S-LLaNA* are trained in two stages: first, the projection network Ψ is trained independently; then, it is

jointly fine-tuned with the LLM. This protocol, common among state-of-the-art MLLMs [74, 133], leads to high computational costs, particularly prohibitive when using large LLMs with tens of billions of parameters. This section investigates whether comparable performance can be achieved without fine-tuning the LLM, by optimizing only the projection network Ψ on the full dataset. Results are shown in Table 4.6, where rows marked "LLM Finetuned" correspond to the two-stage protocol, while the remaining rows represent models trained solely with the first stage on the full dataset. For the brief captioning task on *ObjaNeRF-Text*, we evaluate the models on the *PointLLM* test set, which is larger than the *GPT4Point* test set. Interestingly, *S-LLaNA* maintains excellent performance even without finetuning LLaMA. In fact, the non-finetuned *S-LLaNA-7b* and *S-LLaNA-13b* closely match their finetuned counterparts with performance gaps consistently under 1 point across all metrics. In some cases, the non-finetuned variants even outperform the finetuned ones. For instance, on the brief captioning task of *ShapeNeRF-Text*, the Sentence-BERT score improves from 78.91 to 79.29 for *S-LLaNA-7B*, and from 78.98 to 79.05 for *S-LLaNA-13B*. In contrast, LLaNA struggles without LLM fine-tuning. Its non-finetuned versions exhibit substantial drops across all benchmarks, with gaps exceeding 10 points on challenging tasks like *Spatial ObjaNeRF*. This trend persists even for the larger 13B model. We hypothesize that the failure of LLaNA in the non-finetuned regime stems from the global NeRF representation used as input to the LLM. This representation cannot be directly mapped to a specific language token, making fine-tuning of the LLM — and thus adaptation of its input token space — a crucial step. In contrast, to represent objects, *S-LLaNA* uses $N = 3072$ spatial tokens that encode more local and lower-level concepts. Thus, it is more likely that the projector can learn a mapping from these tokens to the input space of the LLM, without requiring LLM fine-tuning. This property of *S-LLaNA* allows for significantly faster and more cost-effective training, while still achieving comparable performance.

Visualization of attention maps of LLaMA. In Figure 4.11, we visualize the self-attention maps within the LLM of *S-LLaNA-13b* when prompted with questions designed to force the model to focus on specific object parts of a NeRF from the *ObjaNeRF* test set. Specifically, we extract the self-attention maps computed on the input sequence fed into LLaMa-13b, which consists of $N = 3072$ NeRF tokens, concatenated with the text tokens of the question, for a total of B tokens. For visualization, we focus on the final decoder

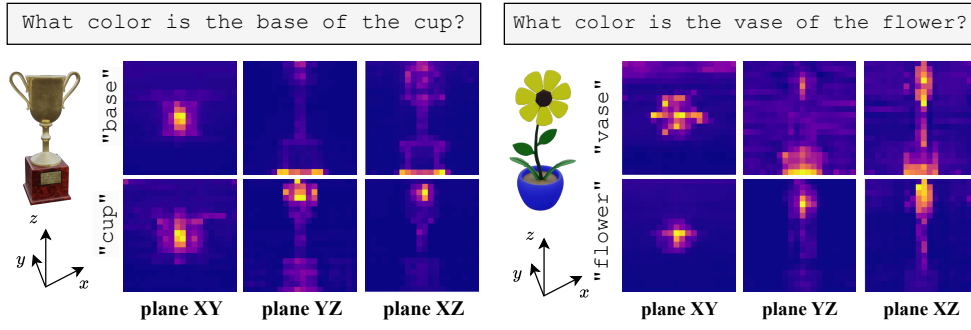


FIGURE 4.11: Visualization of the self-attention maps of the LLM (LLaMA 2)

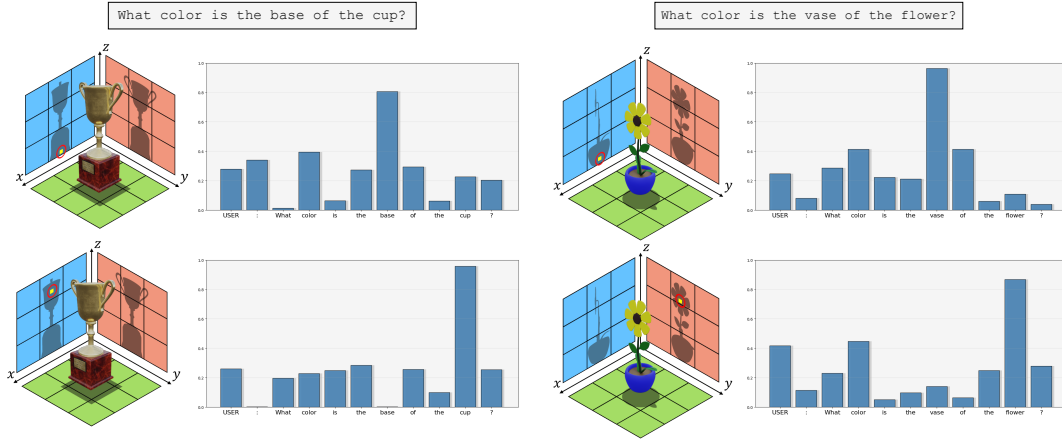


FIGURE 4.12: Visualization of the normalized self-attention scores of single NeRF tokens towards the text tokens of the question. In yellow, surrounded by a red circle, the reference NeRF token.

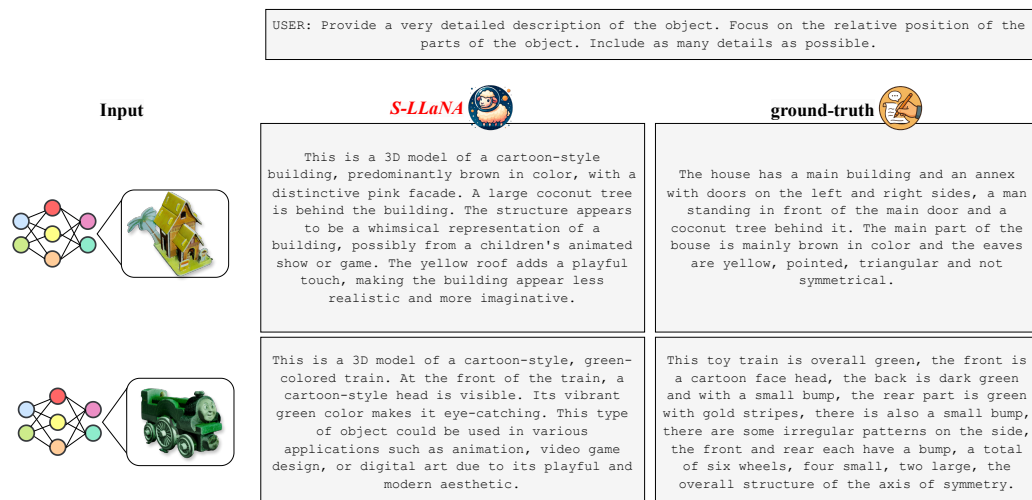
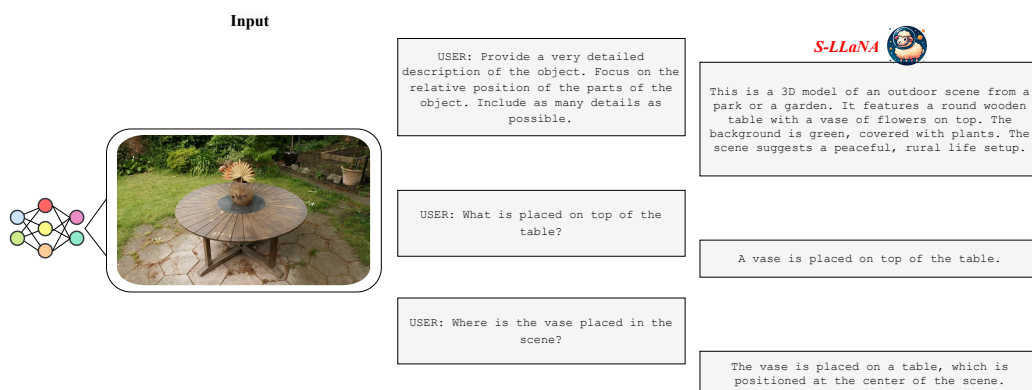
layer of LLaMA-13b and average the attention scores across all 40 attention heads, obtaining a matrix $A \in \mathbb{R}^{B \times B}$. We visualize the part of a column in A corresponding to the attention between a word in the question and the 3072 spatial tokens. To better visualize these scores, we reshape them into three images corresponding to each tri-plane of dimension $H \times W$, where $H = W = 32$. Several patterns emerge from these attention maps. First, in each plane of the tri-plane, the outline of the object is distinctly visible. This indicates that the supervision of the weights2space meta-encoder successfully injects spatial object information into the NeRF token representation. Moreover, attention is clearly localized: tokens with the highest attention scores to a given word correspond to spatial regions where the referenced element is located. For instance, in the right example, the word “flower” activates tokens in the upper region of the object, and “vase” focuses attention on the lower portion. These visualizations highlight the ability of *S-LLaNA* to ground linguistic concepts into spatially localized NeRF representations. In Figure 4.12, we show the self-attention maps in the opposite direction. For a given NeRF

token at a specific object location, we visualize the normalized attention scores to each text token in the question. These maps are derived from the final decoder layer of LLaMA-13B, with attention scores averaged across all 40 heads. These attention maps confirm the ability of the `weights2space` meta-encoder to generate spatially-aware tokens from the weights of the input NeRF. As shown in the first row of the Figure, tokens located at the base of the cup and on the vase of the flower exhibit the strongest attention to the words “base” and “vase”, respectively, while showing much lower scores for the remaining words. Similarly, in the second row, tokens covering the top of the objects correlate most strongly with the words “cup” and “flower”. These visualizations further highlight the ability of *S-LLaNA* to connect the words of the question with the NeRF tokens representing specific parts of the object.

4.6.5 Real-world generalization of *Spatial LLaNA*

Although *ObjaNeRF-Text* already contains 3D models of real scenes, to further test the real-world generalization of *S-LLaNA*, we perform qualitative experiments on *OmniObject3D* [131] and *mip-NeRF360* [12] datasets. The former provides meshes of real-scanned objects, captured in a controlled environment with neutral lighting and background, while *mip-NeRF 360* includes complex real-world unbounded scenes. This is one of the most commonly used datasets for novel view synthesis tasks. Figure 4.13 shows qualitative examples on *OmniObject3D*. Such predictions highlight the generalization ability of *S-LLaNA*, which is able to provide meaningful and spatially-grounded descriptions even for scenes that do not belong to the training distribution of the model. In Figure 4.14, we can notice how *S-LLaNA* has the potential to generalize to unbounded in-the-wild NeRF scenes. Such data are extremely different from the training set of the model, from different points of view. First of all, our training data are purely object-centric with a single large object in the center, while in this case the vase of flowers on the table is quite far from the camera view. Secondly, the 3D models of our training set have a neutral back background, while in-the-wild scenes have very complex backgrounds with rich details. Such features make the task of generalizing to in-the-wild scenes very challenging. Despite this, *S-LLaNA* is able to recognize the main subjects of the scene (“a round wooden table with a vase of flowers on top”) and their spatial relations (“A vase is placed on top of the table”). Interestingly, *S-LLaNA* can also provide some information about the background of the scene (“The background is green, covered with plants.”), which has never seen in any

training example.

FIGURE 4.13: Qualitative examples of *S-LLaNA* on OmniObject3DFIGURE 4.14: Qualitative examples of *S-LLaNA* on Mip-NeRF 360

Chapter 5

Final remarks

5.1 Summary of Thesis Achievements

In this Thesis, novel deep learning techniques have been proposed to contribute to the field of multimodal AI for 3D-language understanding.

In [Chapter 1](#), we introduced the fundamental background of this field, providing the reader with the tools to navigate this manuscript. We described the most common 3D representations in computer vision and provided a review on the state-of-the-art in multimodal 3D-language deep learning.

In [Chapter 2](#), we presented a benchmark for text-to-shape coherence, comprising: *GPT2Shape*, a dataset of 3D models paired with high-quality language annotations, *CrossCoherence*, a state-of-the-art quantitative metric for text-to-3D coherence, suitable for the evaluation of text-driven 3D generative models, and *HST*, a human-validated test set for the evaluation of such models. This benchmark showed impressive performance on this task, outperforming commonly used quantitative metrics that rely on 2D representations.

[Chapter 3](#) described the origin of NeRF-language models, i.e., Multimodal Large Language Models specifically designed to ingest NeRFs as the only input modality. Such models have the key characteristic to directly process the weights of a NeRF’s MLP, allowing to obtain a holistic representation of the input data, without the need to render images or materialize 3D data structures, which are daunting tasks from a computational point of view and sensitive to rendering parameters. The effectiveness of this strategy was confirmed by a wide range of quantitative experiments on ShapeNeRF-Text and ObjaNeRF-Text, the established benchmarks for NeRF-language tasks. Our model *LLaNA* was able to outperform state-of-the-art Multimodal Large Language Models which process explicit data representations, such as images and point clouds.

[Chapter 4](#) introduced a method to compute spatially-aware tokens from the weights of a NeRF’s MLP. This local representation was shown to be extremely

rich in terms of fine-grained detail and spatial understanding, empowering the definition of *Spatial LLaNA*, a new NeRF-language Model capable of performing detailed spatial captioning and Q&A on complex scenes with multiple objects. Our model also showed promising performance on real-world generalization tasks.

5.2 Current limitations

Despite the significant progress reported in this Thesis, multimodal 3D-language understanding and generation remain substantially constrained when compared to the quality and robustness exhibited by recent vision-language models.

A primary limitation stems from the critical scarcity of large-scale, high-quality paired 3D-language datasets. The creation of such datasets is extremely challenging, due to the inherent difficulty in acquiring, processing, and annotating 3D data. Furthermore, the predominance of synthetic datasets limits their domain diversity.

In the specific case of NeRF-language models, research is still in its early phase. Existing approaches rely on datasets of limited size, restricting their scalability and generalization. Furthermore, these models are typically designed around a specific NeRF architecture, limiting their compatibility with the many architectural variants that were published in literature. These limitations slow down the broader adoption of NeRF-language models within the research community.

Finally, as models become increasingly large and multimodal, their training and deployment costs grow substantially. The current reliance on very large language backbones makes experimentation expensive and limits accessibility for smaller research groups. This poses an additional barrier to progress and inclusivity in the field.

These limitations illuminate several promising avenues for future research.

5.3 Future Directions

The future of multimodal 3D-language understanding will be shaped by the creation of large-scale, diverse, and well-annotated datasets. Recent advances in 3D scanning hardware and software make the large-scale digitization of real-world assets increasingly feasible. At the same time, the availability of powerful language and vision-language models offers new opportunities for

generating or refining text annotations automatically. The emergence of such datasets will enable the training of 3D-language foundation models, potentially playing a role analogous to CLIP [99] in the vision-language domain, providing a shared backbone for a wide range of downstream 3D tasks.

Beyond fixed representations: the shift to Gaussian Splatting. While this Thesis explored NeRF-based architectures, a critical area for future research is the adaptation of these findings to 3D Gaussian Splatting (3DGS) [53]. As the community increasingly pivots toward 3DGS for its real-time rendering capabilities, a natural question arises: Can weight-based reasoning (like the `weights2space` approach proposed in this Thesis) translate to splat-based representations?

Unlike the implicit weights of an MLP in a NeRF, 3DGS parameters are explicit and spatially grounded. However, one of the core ideas of this Thesis, i.e. treating the parameters of a neural network as tokens for a Transformer, remains highly applicable. Future work could explore: (a) The adaptation of `weights2space` to map the geometric and appearance attributes of individual Gaussians into a shared latent space; (b) The use of Point-based Transformers as encoders for 3DGS; (b) The development of a unified multimodal framework capable of processing both implicit (NeRF) and explicit (3DGS) representations, thereby enhancing model generalization across diverse data sources.

Task expansion. In addition to the type of representation used, the tasks these models can perform can also be expanded. While the focus of this Thesis was on captioning and question-answering (Q&A) tasks, a logical extension would be the inclusion of 3D grounding. This task would further test and develop the spatial reasoning and fine-grained localization capabilities of NeRF-language models.

From Understanding to Generation. A compelling extension of the findings in this Thesis is the transition from 3D understanding to 3D generation. While *Spatial LLaNA* treats the tri-plane representation as an input for multimodal reasoning, the same discretized latent space could be utilized as a target output for a Large Language Model. Recent literature has demonstrated that LLMs can be transformed into generative models by framing 3D generation as a next-token prediction task [134, 23, 21, 129, 112]. Following this logic,

an LLM could be trained to autoregressively generate the tokens of a tri-plane representation conditioned on a text prompt. Because the tri-plane provides a structured yet compact spatial encoding, it represents a more tractable generation target than raw 3D voxels or unconstrained MLP weights. This "Generative *Spatial LLaNA*" would effectively unify 3D perception and synthesis within a single transformer backbone, enabling a truly bidirectional interface where the model can both describe a 3D scene and manifest one from natural language descriptions.

Efficiency. Finally, future work should also focus on improving the computational efficiency and scalability of these models. Research into more compact, modular architectures or efficient training strategies could make multimodal 3D-language models more tractable and sustainable for broad deployment. Reducing their computational footprint would significantly lower the barrier to entry for researchers and practitioners, accelerate experimentation, and pave the way for real-time applications beyond academic research.

The path towards inclusive, efficient, and truly multimodal artificial intelligence is undoubtedly challenging, yet it has never been more exciting than it is today.

Bibliography

- [1] Josh Achiam et al. “Gpt-4 technical report”. In: *arXiv preprint arXiv:2303.08774* (2023).
- [2] Panos Achlioptas et al. “ShapeGlot: Learning language for shape differentiation”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 8938–8947.
- [3] Abien Fred Agarap. “Deep learning using rectified linear units (relu)”. In: *arXiv preprint arXiv:1803.08375* (2018).
- [4] Andrea Amaduzzi et al. “Llana: Large language and nerf assistant”. In: *Advances in Neural Information Processing Systems* 38 (2024), pp. 1162–1195.
- [5] Andrea Amaduzzi et al. “Looking at words and points with attention: a benchmark for text-to-shape coherence”. In: *2023 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*. IEEE Computer Society. 2023, pp. 2860–2869.
- [6] Andrea Amaduzzi et al. “Scaling LLaNA: Advancing NeRF-Language Understanding Through Large-Scale Training”. In: *arXiv preprint arXiv:2504.13995* (2025).
- [7] Andrea Amaduzzi et al. “Spatially-aware Weights Tokenization for NeRF-Language Models”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2025.
- [8] Haotian Bai et al. “Componerf: Text-guided multi-object compositional nerf with editable 3d scene layout”. In: *arXiv preprint arXiv:2303.13843* (2023).
- [9] Jinze Bai et al. “Qwen-vl: A frontier large vision-language model with versatile abilities”. In: *arXiv preprint arXiv:2308.12966* (2023).
- [10] Francesco Ballerini et al. “Connecting nerfs images and text”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 866–876.

- [11] Satantjeev Banerjee and Alon Lavie. “METEOR: An automatic metric for MT evaluation with improved correlation with human judgments”. In: *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*. 2005, pp. 65–72.
- [12] Jonathan T Barron et al. “Mip-nerf 360: Unbounded anti-aliased neural radiance fields”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 5470–5479.
- [13] Bruce G Baumgart. “A polyhedron representation for computer vision”. In: *Proceedings of the May 19-22, 1975, national computer conference and exposition*. 1975, pp. 589–596.
- [14] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. “Learning long-term dependencies with gradient descent is difficult”. In: *IEEE transactions on neural networks* 5.2 (1994), pp. 157–166.
- [15] Tom Brown et al. “Language models are few-shot learners”. In: *Advances in neural information processing systems* 33 (2020), pp. 1877–1901.
- [16] Adriano Cardace et al. “Neural Processing of Tri-Plane Hybrid Neural Fields”. In: *The Twelfth International Conference on Learning Representations*. 2024. URL: <https://openreview.net/forum?id=zRkM6UcA22>.
- [17] Angel X Chang et al. “Shapenet: An information-rich 3d model repository”. In: *arXiv preprint arXiv:1512.03012* (2015).
- [18] Anpei Chen et al. “TensorRF: Tensorial Radiance Fields”. In: *European Conference on Computer Vision (ECCV)*. 2022.
- [19] Ke Chen et al. “HTS-AT: A hierarchical token-semantic audio transformer for sound classification and detection”. In: *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2022, pp. 646–650.
- [20] Kevin Chen et al. “Text2shape: Generating shapes from natural language by learning joint embeddings”. In: *Asian conference on computer vision*. Springer. 2018, pp. 100–116.
- [21] Sijin Chen et al. *MeshXL: Neural Coordinate Field for Generative 3D Foundation Models*. 2024. arXiv: 2405.20853 [cs.CV].
- [22] Xinlei Chen et al. “Microsoft coco captions: Data collection and evaluation server”. In: *arXiv preprint arXiv:1504.00325* (2015).
- [23] Yongwei Chen et al. “SAR3D: Autoregressive 3D Object Generation and Understanding via Multi-scale 3D VQVAE”. In: *CVPR*. 2025.

- [24] Zilong Chen et al. "Text-to-3d using gaussian splatting". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2024, pp. 21401–21412.
- [25] Wei-Lin Chiang et al. "Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality". In: See <https://vicuna.lmsys.org> (accessed 14 April 2023) 2.3 (2023), p. 6.
- [26] SeungJeh Chung, JooHyun Park, and HyeongYeop Kang. "3dstyleglip: Part-tailored text-guided 3d neural stylization". In: *arXiv preprint arXiv:2404.02634* (2024).
- [27] Kevin Crowston. "Amazon mechanical turk: A research tool for organizations and information systems scholars". In: *Shaping the Future of ICT Research. Methods and Approaches: IFIP WG 8.2, Working Conference, Tampa, FL, USA, December 13-14, 2012. Proceedings*. Springer. 2012, pp. 210–221.
- [28] Wenliang Dai et al. "Instructblip: Towards general-purpose vision-language models with instruction tuning". In: *Advances in Neural Information Processing Systems* 36 (2024).
- [29] Luca De Luigi et al. "Deep Learning on Implicit Neural Representations of Shapes". In: *International Conference on Learning Representations (ICLR)*. 2023.
- [30] Matt Deitke et al. "Objaverse: A universe of annotated 3d objects". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2023, pp. 13142–13153.
- [31] Jia Deng et al. "Imagenet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255.
- [32] Soham Deshmukh et al. "Pengi: An audio language model for audio tasks". In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 18090–18108.
- [33] Jacob Devlin et al. "Bert: Pre-training of deep bidirectional transformers for language understanding". In: *arXiv preprint arXiv:1810.04805* (2018).
- [34] Emilien Dupont et al. "From data to functa: Your data point is a function and you can treat it like one". In: *International Conference on Machine Learning*. PMLR. 2022, pp. 5694–5725.
- [35] Sara Fridovich-Keil et al. "Plenoxels: Radiance fields without neural networks". In: (2022), pp. 5501–5510.

- [36] Rao Fu et al. "Shapecrafter: A recursive text-conditioned 3d shape generation model". In: *arXiv preprint arXiv:2207.09446* (2022).
- [37] Tianyu Gao, Xingcheng Yao, and Danqi Chen. "SimCSE: Simple Contrastive Learning of Sentence Embeddings". In: *2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021*. Association for Computational Linguistics (ACL). 2021, pp. 6894–6910.
- [38] Ross Girshick. "Fast r-cnn". In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1440–1448.
- [39] Ayaan Haque et al. "Instruct-nerf2nerf: Editing 3d scenes with instructions". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2023, pp. 19740–19750.
- [40] Robert Hecht-Nielsen. "On the algebraic structure of feedforward network weight spaces". In: *Advanced Neural Computers*. Elsevier, 1990, pp. 129–135.
- [41] Dan Hendrycks and Kevin Gimpel. "Gaussian error linear units (gelus)". In: *arXiv preprint arXiv:1606.08415* (2016).
- [42] Amir Hertz et al. "Spaghetti: Editing implicit shapes through part aware generation". In: *ACM Transactions on Graphics (TOG)* 41.4 (2022), pp. 1–20.
- [43] Yining Hong et al. "3D-LLM: Injecting the 3D World into Large Language Models". In: *Thirty-seventh Conference on Neural Information Processing Systems*. 2023. URL: <https://openreview.net/forum?id=YQA28p7qNz>.
- [44] Rongjie Huang et al. "AudioGPT: Understanding and generating speech, music, sound, and talking head". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 38. 21. 2024, pp. 23802–23804.
- [45] Shaohan Huang et al. "Language is not all you need: Aligning perception with language models". In: *Advances in Neural Information Processing Systems* 36 (2024).
- [46] Sungwon Hwang et al. "Faceclipnerf: Text-driven 3d face manipulation using deformable neural radiance fields". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 3469–3479.
- [47] Sergey Ioffe and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift". In: *International conference on machine learning*. pmlr. 2015, pp. 448–456.

- [48] Sergey Ioffe and Christian Szegedy. “Batch normalization: Accelerating deep network training by reducing internal covariate shift”. In: *International conference on machine learning*. pmlr. 2015, pp. 448–456.
- [49] Florian Jaeckle and M Pawan Kumar. “Generating adversarial examples with graph neural networks”. In: *Uncertainty in Artificial Intelligence*. PMLR. 2021, pp. 1556–1564.
- [50] Ajay Jain et al. “Zero-shot text-guided object generation with dream fields”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 867–876.
- [51] Kyungmin Jo et al. “Cg-nerf: Conditional generative neural radiance fields for 3d-aware image synthesis”. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2023, pp. 724–733.
- [52] Heewoo Jun and Alex Nichol. “Shap-e: Generating conditional 3d implicit functions”. In: *arXiv preprint arXiv:2305.02463* (2023).
- [53] Bernhard Kerbl et al. “3D Gaussian splatting for real-time radiance field rendering.” In: *ACM Trans. Graph.* 42.4 (2023), pp. 139–1.
- [54] Justin* Kerr et al. “LERF: Language Embedded Radiance Fields”. In: *International Conference on Computer Vision (ICCV)*. 2023.
- [55] Umar Khalid et al. “Latenteditor: Text driven local editing of 3d scenes”. In: *European Conference on Computer Vision*. Springer. 2024, pp. 364–380.
- [56] Boris Knyazev et al. “Parameter Prediction for Unseen Deep Architectures”. In: *Advances in Neural Information Processing Systems*. Ed. by A. Beygelzimer et al. 2021. URL: <https://openreview.net/forum?id=vqHak8NLk25>.
- [57] Sosuke Kobayashi, Eiichi Matsumoto, and Vincent Sitzmann. “Decomposing NeRF for Editing via Feature Field Distillation”. In: *Advances in Neural Information Processing Systems*. Ed. by S. Koyejo et al. Vol. 35. Curran Associates, Inc., 2022, pp. 23311–23330. URL: https://proceedings.neurips.cc/paper_files/paper/2022/file/93f250215e4889119807b6fac3a57aec-Paper-Conference.pdf.
- [58] Miltiadis Kofinas et al. “Graph Neural Networks for Learning Equivariant Representations of Neural Networks”. In: *The Twelfth International Conference on Learning Representations*. 2024.
- [59] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Communications of the ACM* 60.6 (2017), pp. 84–90.

- [60] Han-Hung Lee and Angel X. Chang. *Understanding Pure CLIP Guidance for Voxel Grid NeRF Models*. 2022. arXiv: 2209.15172 [cs.CV].
- [61] Marc Levoy and Turner Whitted. "The use of points as a display primitive". In: (1985).
- [62] Jianhui Li et al. "InstructPix2NeRF: Instructed 3D Portrait Editing from a Single Image". In: *The Twelfth International Conference on Learning Representations*. 2024. URL: <https://openreview.net/forum?id=XIxhINXtQk>.
- [63] Junnan Li et al. "BLIP-2: Bootstrapping language-image pre-training with frozen image encoders and large language models". In: *arXiv preprint arXiv:2301.12597* (2023).
- [64] KunChang Li et al. "Videochat: Chat-centric video understanding". In: *arXiv preprint arXiv:2305.06355* (2023).
- [65] Ruilong Li, Matthew Tancik, and Angjoo Kanazawa. "Nerfacc: A general nerf acceleration toolbox". In: *arXiv preprint arXiv:2210.04847* (2022).
- [66] Yiheng Li et al. "UniPose: A Unified Multimodal Framework for Human Pose Comprehension, Generation and Editing". In: *arXiv preprint arXiv:2411.16781* (2024).
- [67] Zongxia Li et al. "A survey of state of the art large vision language models: Alignment, benchmark, evaluations and challenges". In: *arXiv preprint arXiv:2501.02189* (2025).
- [68] Yixun Liang et al. "UniTEX: Universal High Fidelity Generative Texturing for 3D Shapes". In: *arXiv preprint arXiv:2505.23253* (2025).
- [69] Derek Lim et al. "Graph Metanetworks for Processing Diverse Neural Architectures". In: *The Twelfth International Conference on Learning Representations*. 2024. URL: <https://openreview.net/forum?id=ijK5hyxs0n>.
- [70] Chen-Hsuan Lin et al. "Magic3d: High-resolution text-to-3d content creation". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2023, pp. 300–309.
- [71] Chin-Yew Lin. "Rouge: A package for automatic evaluation of summaries". In: *Text summarization branches out*. 2004, pp. 74–81.
- [72] Tsung-Yi Lin et al. "Microsoft coco: Common objects in context". In: *European conference on computer vision*. Springer. 2014, pp. 740–755.

- [73] Xiongkun Linghu et al. "Multi-modal Situated Reasoning in 3D Scenes". In: *Advances in Neural Information Processing Systems*. Ed. by A. Globerson et al. Vol. 37. Curran Associates, Inc., 2024, pp. 140903–140936. URL: https://proceedings.neurips.cc/paper_files/paper/2024/file/feaeec8ec2d3cb131fe18517ff14ec1f-Paper-Datasets_and_Benchmarks_Track.pdf.
- [74] Haotian Liu et al. "Visual Instruction Tuning". In: *Advances in Neural Information Processing Systems*. Ed. by A. Oh et al. Vol. 36. Curran Associates, Inc., 2023, pp. 34892–34916. URL: https://proceedings.neurips.cc/paper_files/paper/2023/file/6dcf277ea32ce3288914faf369fe6de0-Paper-Conference.pdf.
- [75] Ruyang Liu et al. "BT-Adapter: Video Conversation is Feasible Without Video Instruction Tuning". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 13658–13667.
- [76] Zhengzhe Liu et al. "Towards implicit text-guided 3d shape generation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 17896–17906.
- [77] William E Lorensen and Harvey E Cline. "Marching cubes: A high resolution 3D surface construction algorithm". In: *Seminal graphics: pioneering efforts that shaped the field*. 1998, pp. 347–353.
- [78] Ilya Loshchilov and Frank Hutter. "Decoupled weight decay regularization". In: *arXiv preprint arXiv:1711.05101* (2017).
- [79] Jingyue Lu and M. Pawan Kumar. "Neural Network Branching for Neural Network Verification". In: *International Conference on Learning Representations*. 2020. URL: <https://openreview.net/forum?id=B1evfa4tPB>.
- [80] Tiange Luo et al. "Scalable 3d captioning with pretrained models". In: *Advances in Neural Information Processing Systems* 36 (2024).
- [81] Fan Ma et al. "Vista-llama: Reducing hallucination in video language models via equal distance to visual tokens". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 13151–13160.
- [82] Muhammad Maaz et al. "Video-chatgpt: Towards detailed video understanding via large vision and language models". In: *arXiv preprint arXiv:2306.05424* (2023).

- [83] Ricardo Martin-Brualla et al. “Nerf in the wild: Neural radiance fields for unconstrained photo collections”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 7210–7219.
- [84] Guofeng Mei et al. “PerLA: Perceptive 3D language assistant”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2025.
- [85] Gal Metzer et al. “Latent-nerf for shape-guided generation of 3d shapes and textures”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2023, pp. 12663–12673.
- [86] Oscar Michel et al. “Text2mesh: Text-driven neural stylization for meshes”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 13492–13502.
- [87] Ben Mildenhall et al. “Nerf in the dark: High dynamic range view synthesis from noisy raw images”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 16190–16199.
- [88] Ben Mildenhall et al. “Nerf: Representing scenes as neural radiance fields for view synthesis”. In: *Communications of the ACM* 65.1 (2021), pp. 99–106.
- [89] Ashkan Mirzaei et al. “Reference-guided controllable inpainting of neural radiance fields”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2023, pp. 17815–17825.
- [90] Thomas Müller et al. “Instant Neural Graphics Primitives with a Multiresolution Hash Encoding”. In: *ACM Trans. Graph.* 41.4 (July 2022), 102:1–102:15. DOI: [10.1145/3528223.3530127](https://doi.org/10.1145/3528223.3530127). URL: <https://doi.org/10.1145/3528223.3530127>.
- [91] Aviv Navon et al. “Equivariant Architectures for Learning in Deep Weight Spaces”. In: *International Conference on Machine Learning*. 2023.
- [92] Alex Nichol et al. “Point-e: A system for generating 3d point clouds from complex prompts”. In: *arXiv preprint arXiv:2212.08751* (2022).
- [93] Kishore Papineni et al. “Bleu: a method for automatic evaluation of machine translation”. In: *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*. 2002, pp. 311–318.

- [94] Dong Huk Park et al. "Benchmark for Compositional Text-to-Image Synthesis". In: *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*. Ed. by J. Vanschoren and S. Yeung. Vol. 1. 2021. URL: <https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/file/0a09c8844ba8f0936c20bd791130d6b6-Paper-round1.pdf>.
- [95] Ben Poole et al. "Dreamfusion: Text-to-3d using 2d diffusion". In: *arXiv preprint arXiv:2209.14988* (2022).
- [96] Charles R Qi et al. "Pointnet: Deep learning on point sets for 3d classification and segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 652–660.
- [97] Charles Ruizhongtai Qi et al. "Pointnet++: Deep hierarchical feature learning on point sets in a metric space". In: *Advances in neural information processing systems* 30 (2017).
- [98] Zhangyang Qi et al. "GPT4Point: A Unified Framework for Point-Language Understanding and Generation". In: *CVPR*. 2024.
- [99] Alec Radford et al. "Learning transferable visual models from natural language supervision". In: *International conference on machine learning*. PmLR. 2021, pp. 8748–8763.
- [100] Colin Raffel et al. "Exploring the limits of transfer learning with a unified text-to-text transformer". In: *The Journal of Machine Learning Research* 21.1 (2020), pp. 5485–5551.
- [101] Aditya Ramesh et al. "Zero-shot text-to-image generation". In: *International conference on machine learning*. Pmlr. 2021, pp. 8821–8831.
- [102] Pierluigi Zama Ramirez et al. "Deep Learning on Object-centric 3D Neural Fields". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2024).
- [103] Nils Reimers and Iryna Gurevych. "Sentence-bert: Sentence embeddings using siamese bert-networks". In: *arXiv preprint arXiv:1908.10084* (2019).
- [104] Elad Richardson et al. "Texture: Text-guided texturing of 3d shapes". In: *ACM SIGGRAPH 2023 conference proceedings*. 2023, pp. 1–11.
- [105] Robin Rombach et al. "High-resolution image synthesis with latent diffusion models". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 10684–10695.

- [106] Chitwan Saharia et al. "Photorealistic text-to-image diffusion models with deep language understanding". In: *Advances in neural information processing systems* 35 (2022), pp. 36479–36494.
- [107] Aditya Sanghi et al. "Clip-forge: Towards zero-shot text-to-shape generation". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 18603–18613.
- [108] Christoph Schuhmann et al. "Laion-5b: An open large-scale dataset for training next generation image-text models". In: *Advances in neural information processing systems* 35 (2022), pp. 25278–25294.
- [109] Konstantin Schürholt, Dimche Kostadinov, and Damian Borth. "Self-Supervised Representation Learning on Neural Network Weights for Model Characteristic Prediction". In: *Advances in Neural Information Processing Systems*. Ed. by A. Beygelzimer et al. 2021. URL: <https://openreview.net/forum?id=F1D8buayXQT>.
- [110] Bipasha Sen et al. "HyP-NeRF: Learning Improved NeRF Priors using a HyperNetwork". In: *Advances in Neural Information Processing Systems*. Ed. by A. Oh et al. Vol. 36. Curran Associates, Inc., 2023, pp. 51050–51064. URL: https://proceedings.neurips.cc/paper_files/paper/2023/file/a03037317560b8c5f2fb4b6466d4c439-Paper-Conference.pdf.
- [111] Hoigi Seo et al. *DITTO-NeRF: Diffusion-based Iterative Text To Omnidirectional 3D Model*. 2023. arXiv: 2304.02827 [cs.CV].
- [112] Yawar Siddiqui et al. "MeshGPT: Generating Triangle Meshes with Decoder-Only Transformers". In: *arXiv preprint arXiv:2311.15475* (2023).
- [113] Oleksii Sidorov et al. "Textcaps: a dataset for image captioning with reading comprehension". In: *European conference on computer vision*. Springer. 2020, pp. 742–758.
- [114] Hyeonseop Song et al. "Blending-nerf: Text-driven localized editing in neural radiance fields". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2023, pp. 14383–14393.
- [115] Matteo Stefanini et al. "From show to tell: A survey on deep learning-based image captioning". In: *IEEE transactions on pattern analysis and machine intelligence* 45.1 (2022), pp. 539–559.
- [116] Chen Sun et al. "Revisiting unreasonable effectiveness of data in deep learning era". In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 843–852.

- [117] Cheng Sun, Min Sun, and Hwann-Tzong Chen. "Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 5459–5469.
- [118] Chunyi Sun et al. "Nerfeditor: Differentiable style decomposition for 3d scene editing". In: *Proceedings of the IEEE/CVF winter conference on applications of computer vision*. 2024, pp. 7306–7315.
- [119] Jiaxiang Tang et al. "Dreamgaussian: Generative gaussian splatting for efficient 3d content creation". In: *arXiv preprint arXiv:2309.16653* (2023).
- [120] Ayush Tewari et al. "State of the art on neural rendering". In: *Computer graphics forum*. Vol. 39. 2. Wiley Online Library. 2020, pp. 701–727.
- [121] Bart Thomee et al. "YFCC100M: The new data in multimedia research". In: *Communications of the ACM* 59.2 (2016), pp. 64–73.
- [122] Hugo Touvron et al. "Llama: Open and efficient foundation language models". In: *arXiv preprint arXiv:2302.13971* (2023).
- [123] Christina Tsalicoglou et al. "Textmesh: Generation of realistic 3d meshes from text prompts". In: *arXiv preprint arXiv:2304.12439* (2023).
- [124] Michael Tschannen et al. "Image Captioners Are Scalable Vision Learners Too". In: *arXiv preprint arXiv:2306.07915* (2023).
- [125] Thomas Unterthiner et al. "Predicting Neural Network Accuracy from Weights". In: *arXiv abs/2002.11448* (2020).
- [126] Ashish Vaswani et al. "Attention is All you Need". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.
- [127] Can Wang et al. "Clip-nerf: Text-and-image driven manipulation of neural radiance fields". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 3835–3844.
- [128] Can Wang et al. "NeRF-Art: Text-Driven Neural Radiance Fields Stylization". In: *IEEE Transactions on Visualization and Computer Graphics* (2023), pp. 1–15. DOI: [10.1109/TVCG.2023.3283400](https://doi.org/10.1109/TVCG.2023.3283400).
- [129] Zhengyi Wang et al. *LLaMA-Mesh: Unifying 3D Mesh Generation with Language Models*. 2024. arXiv: [2411.09595](https://arxiv.org/abs/2411.09595) [cs.LG]. URL: <https://arxiv.org/abs/2411.09595>.

- [130] Zhengyi Wang et al. "Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation". In: *Advances in neural information processing systems* 36 (2023), pp. 8406–8441.
- [131] Tong Wu et al. "Omniobject3d: Large-vocabulary 3d object dataset for realistic perception, reconstruction and generation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 803–814.
- [132] Haomiao Xiong et al. *3UR-LLM: An End-to-End Multimodal Large Language Model for 3D Scene Understanding*. 2025. arXiv: 2501.07819 [cs.CV]. URL: <https://arxiv.org/abs/2501.07819>.
- [133] Runsen Xu et al. "Pointllm: Empowering large language models to understand point clouds". In: *arXiv preprint arXiv:2308.16911* (2023).
- [134] Junliang Ye et al. "ShapeLLM-Omni: A Native Multimodal LLM for 3D Generation and Understanding". In: *arXiv preprint arXiv:2506.01853* (2025).
- [135] Lin Yen-Chen et al. "Nerf-supervision: Learning dense object descriptors from neural radiance fields". In: *2022 international conference on robotics and automation (ICRA)*. IEEE. 2022, pp. 6496–6503.
- [136] Xumin Yu et al. "Point-bert: Pre-training 3d point cloud transformers with masked point modeling". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 19313–19322.
- [137] Yingchen Yu et al. "MorphNeRF: Text-Guided 3D-aware Editing via Morphing Generative Neural Radiance Fields". In: *IEEE Transactions on Multimedia* (2024), pp. 1–13. DOI: [10.1109/TMM.2024.3379888](https://doi.org/10.1109/TMM.2024.3379888).
- [138] Mert Yuksekgonul et al. "When and Why Vision-Language Models Behave like Bags-Of-Words, and What to Do About It?" In: *The Eleventh International Conference on Learning Representations*. 2022.
- [139] Qi Zekun et al. "Shapellm: Universal 3d object understanding for embodied interaction". In: *European Conference on Computer Vision*. Springer. 2024, pp. 214–238.
- [140] Chenshuang Zhang et al. "Text-to-image diffusion models in generative ai: A survey. arxiv 2023". In: *arXiv preprint arXiv:2303.07909* (2023), pp. 423–438.
- [141] Dong Zhang et al. "Speechgpt: Empowering large language models with intrinsic cross-modal conversational abilities". In: *arXiv preprint arXiv:2305.11000* (2023).

- [142] Hang Zhang, Xin Li, and Lidong Bing. "Video-llama: An instruction-tuned audio-visual language model for video understanding". In: *arXiv preprint arXiv:2306.02858* (2023).
- [143] Allan Zhou, Chelsea Finn, and James Harrison. "Universal Neural Functionals". In: *arXiv preprint arXiv:2402.05232* (2024).
- [144] Allan Zhou et al. "Permutation equivariant neural functionals". In: *Advances in neural information processing systems* 37 (2023).
- [145] Jingyu Zhuang et al. "Dreameditor: Text-driven 3d scene editing with neural fields". In: *SIGGRAPH Asia 2023 Conference Papers*. 2023, pp. 1–10.
- [146] Qi Zuo et al. "High-Fidelity 3D Textured Shapes Generation by Sparse Encoding and Adversarial Decoding". In: *European Conference on Computer Vision*. 2024.