Spatially-aware Weights Tokenization for NeRF-Language Models

Andrea Amaduzzi
andrea.amaduzzi4@unibo.it

Pierluigi Zama Ramirez
pierluigi.zama@unibo.it

Giuseppe Lisanti
giuseppe.lisanti@unibo.it

Samuele Salti
samuele.salti@unibo.it

Luigi Di Stefano
luigi.distefano@unibo.it

CVLab, University of Bologna

https://andreamaduzzi.github.io/spatial-llana

Abstract

Neural Radiance Fields (NeRFs) are neural networks – typically multilayer perceptrons (MLPs) – that represent the geometry and appearance of objects, with applications in vision, graphics, and robotics. Recent works propose understanding NeRFs with natural language using Multimodal Large Language Models (MLLMs) that directly process the weights of a NeRF's MLP. However, these approaches rely on a global representation of the input object, making them unsuitable for spatial reasoning and fine-grained understanding. In contrast, we propose `weights2space`, a self-supervised framework featuring a novel meta-encoder that can compute a sequence of spatial tokens directly from the weights of a NeRF. Leveraging this representation, we build *Spatial LLaNA*, a novel MLLM for NeRFs, capable of understanding details and spatial relationships in objects represented as NeRFs. We evaluate *Spatial LLaNA* on NeRF captioning and NeRF Q&A tasks, using both existing benchmarks and our novel Spatial ObjaNeRF dataset consisting of 100 manually-curated language annotations for NeRFs. This dataset features 3D models and descriptions that challenge the spatial reasoning capability of MLLMs. *Spatial LLaNA* outperforms existing approaches across all tasks.

1 Introduction

Neural Radiance Fields (NeRFs) [45] can encode the 3D geometry and photorealistic appearance of an object compactly within the weights of a neural network, typically implemented as a Multi-Layer Perceptron (MLP). The ability of NeRFs to capture intricate visual details has led to widespread adoption in vision, graphics, and robotics [21, 59].

With NeRFs emerging as an effective and popular modality to represent 3D objects, what if one could effortlessly interact through words with an AI assistant capable of providing detailed information about a complex object stored in a computer as a NeRF? For instance, one may ask the assistant to describe specific parts of the object or to explain intricate spatial relationships precisely. Such intuitive, language-based interaction with NeRF representations has recently been investigated by LLaNA [6, 7]. LLaNA employs a meta-encoder [53] to extract a global embedding of an object

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

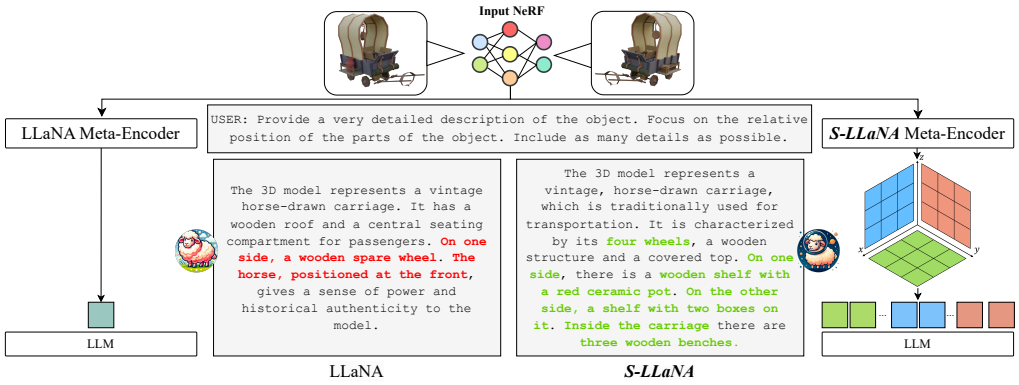


Figure 1: (left) LLaNA relies on a single global representation of the input NeRF. (right) *S-LLaNA* processes multiple spatially-aware tokens extracted by our new meta-encoder and, thus, can provide detailed descriptions that capture the relationships between objects. In particular, the answer of *Spatial 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"*).

stored as a NeRF by directly processing the weigts of the MLP. Compared to rendering views from the given NeRF so as to feed them to a Vision-Language Model, the strategy introduced by LLaNA is remarkably faster and avoids decisions that impact object understanding, such as the number of views to render and their resolution. The global representation computed by the meta-encoder is then projected into the embedding space of a Large Language Model (LLM) and prepended to a language instruction to realize NeRF-Language reasoning. However, we reckon that reliance on a single token to represent a whole NeRF limits the ability of LLaNA to capture fine-grained details and understand precise spatial relationships among objects, as illustrated in Figure 1, left.

Multimodal Large Language Models (MLLMs) that operate on explicit spatial representations, such as images [4, 8, 15, 32, 33, 39, 63, 70] and point clouds [49, 50, 62], typically achieve fine-grained spatial reasoning by dividing inputs into localized regions – images into patches and point clouds into sets – in order to extract localized, spatially-aware tokens that capture geometric and visual context. However, the weights of a NeRF inherently lack such explicit spatial structure, as the spatial information is distributed across all of them, without any weight being directly associated with any specific object part. Consequently, traditional spatially-aware tokenization approaches are not directly applicable to NeRF representations.

This observation raises an intriguing research question: *Is it possible to elicit spatially-aware representations from the information distributed across the weights of a NeRF?*

To answer this question, 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 the MLP weights and outputs a sequence of tokens that are reorganized into a *tri-plane* [13] 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, capable of reasoning about fine-grained object details and spatial relationships, as shown in Figure 1, right.

To validate our approach, we thoroughly evaluate *Spatial LLaNA* on the existing NeRF-Language datasets: ShapeNeRF-Text [53], HST [5], and ObjaNeRF-Text [7]. Moreover, we propose a novel benchmark, *Spatial ObjaNeRF*, consisting of 100 NeRF models from Objaverse [18] paired with detailed language descriptions focusing on object parts and their spatial relations. Experimental results show a substantial improvement in NeRF-Language reasoning with our approach.

In summary, we make the following contributions:

• *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 ObjaNeRF*: 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 the new spatial reasoning tasks of our dataset as well as on existing benchmarks for NeRF captioning and NeRF Q&A, demonstrating the effectiveness of our approach.

2 Related work

Deep learning on neural networks. Processing the weights of a neural network poses distinct challenges compared to standard input formats, primarily due to the high dimensionality of the weight space, inherent symmetries [23], and the influence of randomness on the convergence point of training [3, 20]. Early approaches to meta-networks use group theory to design architectures equivariant to permutation symmetries in network weights [46, 69], but were limited to specific architectures like MLPs or CNNs. To generalize across architectures, Graph Meta-Networks (GMNs) were introduced [31, 37], reframing the problem as one of converting neural networks into graphs, which GMNs can process due to their inherent permutation equivariance. nf2vec [53] is the first method to perform tasks on NeRFs by directly processing its weights. This model computes a global embedding from the weights of the NeRF's MLP, leveraging an encoder-decoder architecture supervised with a rendering loss. LLaNA [6] uses the feature vector computed by this encoder to perform NeRF-Language tasks, such as NeRF captioning and NeRF Q&A. While nf2vec is designed to ingest MLPs, [11] proposes a method to process a NeRF architecture consisting of a tri-plane structure,performing the tasks of classification and segmentation on NeRF data. All these methods compute a single feature vector from the weights of the input NeRF, encapsulating global information about the object. In contrast, the weights2space meta-encoder proposed in this work extracts spatially-aware tokens from the weights of the input NeRF, providing a richer set of information.

Multimodal Large Language Models. Multimodal Large Language Models (MLLMs) expand the capabilities of existing LLMs [1, 29, 52, 56, 68] by combining language with other modalities like images, audio, or 3D data. Vision-language models [4, 8, 15, 32, 33, 39, 63, 70] encode images using pre-trained Vision Transformers (ViT) [51] that convert image patches into sequences of visual tokens. These local representations capture fine-grained spatial information across the image. 3D MLLMs [25, 49, 50, 62] divide the input point cloud into local patches. Transformer-based encoders like Point-BERT [64] and Recon++ [49] compute discrete token sequences that preserve spatial relationships. MLLMs for audio [19, 26, 65] split acoustic inputs into temporal patches and encode them using audio-specific encoders [14], producing sequences of tokens that retain temporal structure. Similarly, MLLMs for video-language understanding [34, 41, 43, 44, 66] use visual encoders to compute local features on each video frame, preserving both spatial and temporal information. In contrast to these local representation approaches, the only existing MLLM for Neural Radiance Fields (NeRFs), LLaNA [6], 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 *S-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.

Language and Neural Radiance Fields. Given a textual description of an object, generative models produce realistic objects represented as NeRFs [28, 48, 60, 67]. Similarly, other approaches focus on editing an input scene represented as a NeRF, through text control [17, 55, 57, 58]. The interaction between NeRF and language is also leveraged for 3D semantic understanding. On this task, LeRF [30] predicts language features for each spatial location alongside density and color. While the primary focus of LeRF is on object localization, 3D-OVS [40] distills the reasoning capability of CLIP [51]

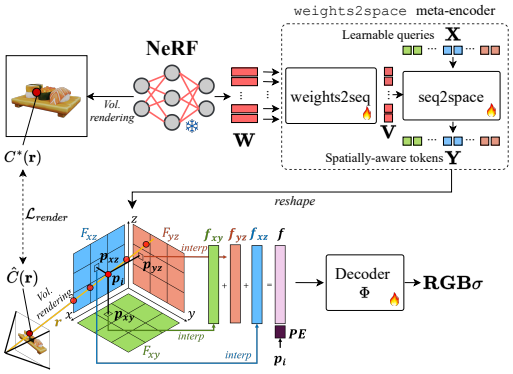


Figure 2: `weights2space` **overview.**

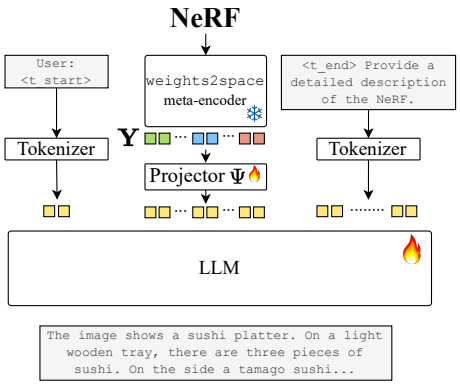


Figure 3: ***Spatial LLaNA* overview.**

and DINO [12] into a NeRF, to perform 3D segmentation. However, these methods train a semantic neural field, relying on the noisy and view-inconsistent semantics from CLIP. In contrast to this methods, OV-NeRF [36] addresses the task of open-vocabulary 3D segmentation by leveraging a regularization technique to improve the single-view semantics and a self-enhancement strategy for cross-view semantics. The neural field adopted by these methods is parametrized by a neural network. The first method to perform captioning and Q&A tasks on NeRF is LLaNA [6] while [7] expands the capabilities of LLaNA to a larger dataset, ObjaNeRF-Text, by scaling its underlying LLM.

3 `weights2space`: spatially-aware NeRF weights tokenization

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 [6], 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 [49, 51, 64]. 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 2 – more implementation details are in the Appendix.

NeRF encoding. Similarly to LLaNA [6], 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 [45] 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 $\mathbf{b}_l \in R^H$. 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` [53] 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.

Meta-encoder. It 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 [27] and ReLU activations [2], 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` [53], 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 [6] 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 [51, 64] 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 $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N) \in \mathbb{R}^{N \times C}$, where N is the number of queries and C is the feature dimension. The Transformer decoder, hence, outputs a new sequence of tokens $\mathbf{Y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N) \in \mathbb{R}^{N \times C}$. 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 [13], 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. 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 $\mathbf{f} \in \mathbb{R}^C$. The detailed explanation of this step is reported in the Appendix.

This feature vector is then concatenated with a positional encoding [45] 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). $\mathbf{\Phi}$ consists of 4 linear layers as the decoder used in [53].

Training. All modules of `weights2space` are optimized end-to-end using differentiable volumetric rendering [35]. 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:

$$\mathcal{L}_{\text{render}} = \sum_{\mathbf{r} \in \mathcal{R}} \text{SmoothL1}(\hat{C}(\mathbf{r}) - C^*(\mathbf{r})) \tag{1}$$

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 ObjaNeRF-Text for 500 epochs with a learning rate of 0.00001, as done in LLaNA [7]. The training needs $\sim$2 days on four 64GB A100 GPUs to reach convergence.

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

Architecture. *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. Inspired by [6], we process these tokens with a shared projector network Ψ, composed of a stack of 2 trainable linear layers, interleaved with $GeLU$ activation functions [24]. This network maps each NeRF token into the internal embedding space of the language model, LLaMA-2 [56]. The input to the LLM consists of the projected NeRF tokens, encapsulated between two learnable special tokens `<t_start>` and `<t_end>`, and concatenated with a sequence of k language tokens corresponding to the user's prompt.

The language model LLaMA processes the combined sequence to autoregressively generate relevant textual output.

Training. We adopt the training protocol introduced in [7], using the ShapeNeRF-Text and ObjaNeRF-Text datasets. In the first stage, *S-LLaNA* is trained using brief textual descriptions, optimizing only the projection layers Ψ and the trainable special tokens `<t_start>` and `<t_end>`. In the 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:

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

where $P(w_t \mid \hat{w}_0 ..., \hat{w}_{t-2}, \hat{w}_{t-1}, \Psi(\mathbf{Y}))$ 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 NeRF token sequence $\Psi(\mathbf{Y})$.

Implementation details. *S-LLaNA* shares the same LLM backbone as LLaNA [7], i.e. the 7B Vicuna and 13B Vicuna [16] of LLaMA-2 [56]. For experimental fairness, *S-LLaNA* follows the same training protocol as [7] and is trained on 300K samples from ShapeNeRF-Text and ObjaNeRF-Text for 3 epochs for each training stage. The projection network $\mathbf{\Psi}$ 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.

Ablation experiments on the tri-plane resolution and tokens dimensionality can be found in the Appendix.

5 Experimental setup

Baselines. The only direct competitor of *S-LLaNA* is LLaNA [7], 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 evaluate also the same reference baselines as LLaNA [7], i.e., state-of-the-art MLLMs ingesting images or 3D explicit 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 [50], PointLLM [62], 3D-LLM [25] and ShapeLLM [49]. As for 2D MLLMs, we consider LLaVA-1.6 [39] and BLIP-2 [33]. 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 [7, 50, 62]: Sentence-BERT [54] and SimCSE [22] based on pre-trained encoders; BLEU-1 [47], ROUGE-L [38], and METEOR [9] based on n-gram statistics.

NeRF-Language datasets. As in [7], 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 models. Results on these datasets are divided into three tasks: brief captioning, detailed captioning, and single-round Q&A conversations. In addition, within the brief captioning task, we evaluate the performance the methods on HST [5], a subset of ShapeNeRF-Text objects with manually annotated short captions.

Spatial ObjaNeRF. To rigorously evaluate the ability of NeRF-Language models to move beyond simple object recognition and capture complex spatial relationships, we introduce Spatial ObjaNeRF, a manually curated benchmark. This dataset consists of 100 human-annotated 3D models from ObjaNeRF-Text. Crucially, to fully assess the spatial reasoning capability of Multi-modal Large Language Models (MLLMs) within a realistic context, every data sample represents a complex scene featuring an arrangement of multiple interacting objects. For each scene, we created a detailed textual

Table 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)

		ShapeNeRF-Text					HST				
Model	**Modality**	**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 2: **NeRF brief captioning on ObjaNeRF-Text.** Best results are in **bold**, runner-up is underlined. (RV: random view, MV: multi-view)

		ObjaNeRF-Text (PointLLM test set)					ObjaNeRF-Text (GPT4Point test set)				
Model	**Modality**	**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
S-LLaNA-7b	NeRF	45.23	45.80	15.72	19.98	**17.70**	50.12	49.55	24.49	**30.72**	26.15
LLaNA-13b	NeRF	42.08	42.40	13.86	17.51	16.18	44.26	43.75	20.61	25.62	22.36
S-LLaNA-13b	NeRF	**45.44**	**46.18**	**15.78**	**20.17**	17.61	**50.68**	**50.30**	**24.53**	30.52	**26.41**

description that emphasizes spatial structure, highlighting the size, shape, and relative positioning of specific objects.

Examples of such annotations and their comparison with those present in ObjaNeRF-Text are provided in the Appendix.

6 Experiments

Results on ShapeNeRF-Text, HST, and ObjaNeRF-Text. We evaluate *S-LLaNA* on the tasks of NeRF brief captioning, detailed captioning, and Q&A, using the datasets ShapeNeRF-Text, HST, and ObjaNeRF-Text. Results are presented in Tables 1 to 4. Across all benchmarks and tasks, *S-LLaNA* consistently outperforms LLaNA and all other 2D and 3D MLLM baselines, often by substantial margins. The 13B variant of *S-LLaNA* 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 1 and 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 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 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

Table 3: **NeRF detailed captioning on ShapeNeRF–Text.** Best results are in **bold**, runner-up is underlined. (FV: front-view, BV: back-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
S-LLaNA-7b	NeRF	**81.03**	**83.00**	44.74	40.08	36.03
LLaNA-13b	NeRF	75.51	77.63	19.87	32.93	20.46
S-LLaNA-13b	NeRF	80.98	82.90	**44.78**	**40.09**	**36.24**

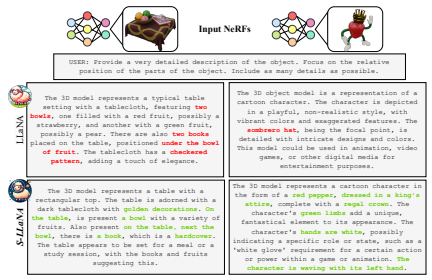
Table 4: **NeRF single-round Q&A on ShapeNeRF–Text.** Best results are in **bold**, runner-up is underlined. (FV: front-view, BV: back-view, MV: multi-view)

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
S-LLaNA-7b	NeRF	83.49	84.06	**50.89**	**58.39**	54.52
LLaNA-13b	NeRF	81.05	81.60	46.08	53.44	49.98
S-LLaNA-13b	NeRF	**83.60**	**84.17**	50.86	58.25	**54.58**

Table 5: **Detailed spatial captioning on Spatial ObjaNeRF.** 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 (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	30.85	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	75.01	77.30	29.43	28.85	21.70
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**

Figure 4: **Qualitative examples on Spatial ObjaNeRF**



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. Additional qualitative examples are in the Appendix.

Results 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 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 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 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 are in the Appendix, together with experiments on the generalization capabilities of *S-LLaNA*.

Ablation on LLM finetuning. As described in Section 5, both LLaNA and *S-LLaNA* are trained in two stages: first, the projection network $\mathbf{\Psi}$ is trained independently; then, it is jointly fine-

Table 6: **Experiments on LLM finetuning.** Best results are in **bold**, runner-up is underlined. (SN-T: ShapeNeRF-Text, ON-T: ObjaNeRF-Text PointLLM test set)

			SN-T brief captioning		ON-T brief captioning		SN-T detailed captioning		SN-T Q&A		Spatial test	
Model	**LLM Finetuned**	**# Trainable params**	**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

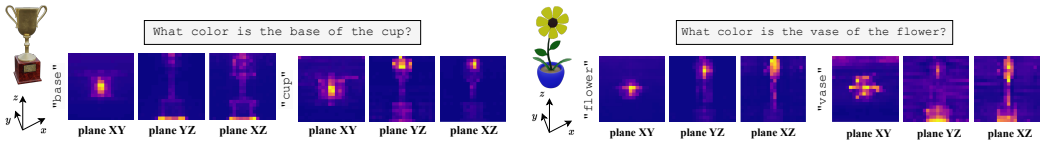


Figure 5: Visualization of the self-attention maps of the LLM (LLaMA 2)

tuned with the LLM. This protocol, common among state-of-the-art MLLMs [39, 62], 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 $\mathbf{\Psi}$ on the full dataset. Results are shown in Table 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 fine-tuned 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 LLM's input space without requiring LLM fine-tuning. This property of *S-LLaNA* allows for significantly faster and more cost-effective training, while still achieving comparable performance. In the Appendix, a discussion on the general language understanding of our model is proposed.

Visualization of attention maps in LLaMA. In Figure 5, 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 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.

7 Conclusions

In this work, we demonstrated that extracting spatially-aware tokens directly from the weights of a NeRF's MLP is possible, enabling fine-grained language interaction on NeRF data. To achieve this goal, we introduced `weights2space`, a self-supervised framework featuring a novel meta-encoder that computes a sequence of spatial tokens, and *S-LLaNA*, a novel Multimodal Large Language Model for NeRF. Our model significantly outperforms existing methods on NeRF-Language tasks, whether they operate directly on NeRFs or use explicit representations. Notably, *S-LLaNA* achieves strong performance even without finetuning its underlying LLM, making it extremely efficient to train. Moreover, we propose Spatial ObjaNeRF, a benchmark designed to evaluate fine-grained spatial understanding in NeRF-Language tasks. Finally, we show qualitatively that our approach leads to interpretable and meaningful attention patterns. This work contributes to the nascent field of Multimodal Large Language Models for Radiance Fields.

Limitations and Future work. Our spatially-aware representation may have the potential to address additional tasks, such as spatial Q&A on 3D scenes, 3D grounding, and 3D object detection. At present, the scarcity of large-scale NeRF-text datasets limits progress in these directions. Expanding this representation to such tasks remains a promising direction for future research.

Acknowledgements

We acknowledge ISCRA for awarding this project access to the LEONARDO supercomputer, owned by the EuroHPC Joint Undertaking, hosted by CINECA (Italy).

This work was partially funded by "FSE+ 2021-2027 ai sensi dell'art. 24, comma 3, lett. a), della Legge 240/2010 e s.m.i. e del D.G.R. 693/2023 (RIF. PA: 2023-20090/RER - CUP: J19J23000730002)".

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Abien Fred Agarap. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*, 2018.

[3] Samuel K Ainsworth, Jonathan Hayase, and Siddhartha Srinivasa. Git re-basin: Merging models modulo permutation symmetries. *arXiv preprint arXiv:2209.04836*, 2022.

[4] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*, 35:23716–23736, 2022.

[5] Andrea Amaduzzi, Giuseppe Lisanti, Samuele Salti, and Luigi Di Stefano. 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)*, pages 2860–2869. IEEE Computer Society, 2023.

[6] Andrea Amaduzzi, Pierluigi Zama Ramirez, Giuseppe Lisanti, Samuele Salti, and Luigi Di Stefano. Llana: Large language and nerf assistant. *Advances in Neural Information Processing Systems*, 37:1162–1195, 2024.

[7] Andrea Amaduzzi, Pierluigi Zama Ramirez, Giuseppe Lisanti, Samuele Salti, and Luigi Di Stefano. Scaling llana: Advancing nerf-language understanding through large-scale training. *arXiv preprint arXiv:2504.13995*, 2025.

[8] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.

[9] Satanjeev 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*, pages 65–72, 2005.

[10] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5470–5479, 2022.

[11] Adriano Cardace, Pierluigi Zama Ramirez, Francesco Ballerini, Allan Zhou, Samuele Salti, and Luigi Di Stefano. Neural processing of tri-plane hybrid neural fields. *arXiv preprint arXiv:2310.01140*, 2023.

[12] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021.

[13] Eric R Chan, Connor Z Lin, Matthew A Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas J Guibas, Jonathan Tremblay, Sameh Khamis, et al. Efficient geometry-aware 3d generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16123–16133, 2022.

[14] Ke Chen, Xingjian Du, Bilei Zhu, Zejun Ma, Taylor Berg-Kirkpatrick, and Shlomo Dubnov. 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)*, pages 646–650. IEEE, 2022.

[15] Xiaokang Chen, Zhiyu Wu, Xingchao Liu, Zizheng Pan, Wen Liu, Zhenda Xie, Xingkai Yu, and Chong Ruan. Janus-pro: Unified multimodal understanding and generation with data and model scaling. *arXiv preprint arXiv:2501.17811*, 2025.

[16] Wei-Lin Chiang, Zhuohan Li, Ziqing Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*, 2(3):6, 2023.

[17] Peng Dai, Feitong Tan, Xin Yu, Yifan Peng, Yinda Zhang, and Xiaojuan Qi. Go-nerf: Generating objects in neural radiance fields for virtual reality content creation. *IEEE Transactions on Visualization and Computer Graphics*, 2025.

[18] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13142–13153, 2023.

[19] Soham Deshmukh, Benjamin Elizalde, Rita Singh, and Huaming Wang. Pengi: An audio language model for audio tasks. *Advances in Neural Information Processing Systems*, 36:18090–18108, 2023.

[20] Rahim Entezari, Hanie Sedghi, Olga Saukh, and Behnam Neyshabur. The role of permutation invariance in linear mode connectivity of neural networks. *arXiv preprint arXiv:2110.06296*, 2021.

[21] Kyle Gao, Yina Gao, Hongjie He, Dening Lu, Linlin Xu, and Jonathan Li. Nerf: Neural radiance field in 3d vision, a comprehensive review. *arXiv preprint arXiv:2210.00379*, 2022.

[22] 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*, pages 6894–6910. Association for Computational Linguistics (ACL), 2021.

[23] Robert Hecht-Nielsen. On the algebraic structure of feedforward network weight spaces. In *Advanced Neural Computers*, pages 129–135. Elsevier, 1990.

[24] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

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

[26] Rongjie Huang, Mingze Li, Dongchao Yang, Jiatong Shi, Xuankai Chang, Zhenhui Ye, Yuning Wu, Zhiqing Hong, Jiawei Huang, Jinglin Liu, et al. Audiogpt: Understanding and generating speech, music, sound, and talking head. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 23802–23804, 2024.

[27] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr, 2015.

[28] Kyungmin Jo, Gyumin Shim, Sanghun Jung, Soyoung Yang, and Jaegul Choo. 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*, pages 724–733, 2023.

[29] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of naacL-HLT*, page 2. Minneapolis, Minnesota, 2019.

[30] Justin Kerr, Chung Min Kim, Ken Goldberg, Angjoo Kanazawa, and Matthew Tancik. Lerf: Language embedded radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19729–19739, 2023.

[31] Miltiadis Kofinas, Boris Knyazev, Yan Zhang, Yunlu Chen, Gertjan J Burghouts, Efstratios Gavves, Cees GM Snoek, and David W Zhang. Graph neural networks for learning equivariant representations of neural networks. *arXiv preprint arXiv:2403.12143*, 2024.

[32] Feng Li, Renrui Zhang, Hao Zhang, Yuanhan Zhang, Bo Li, Wei Li, Zejun Ma, and Chunyuan Li. Llava-next-interleave: Tackling multi-image, video, and 3d in large multimodal models. *arXiv preprint arXiv:2407.07895*, 2024.

[33] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. BLIP-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv preprint arXiv:2301.12597*, 2023.

[34] KunChang Li, Yinan He, Yi Wang, Yizhuo Li, Wenhai Wang, Ping Luo, Yali Wang, Limin Wang, and Yu Qiao. Videochat: Chat-centric video understanding. *arXiv preprint arXiv:2305.06355*, 2023.

[35] Ruilong Li, Matthew Tancik, and Angjoo Kanazawa. Nerfacc: A general nerf acceleration toolbox. *arXiv preprint arXiv:2210.04847*, 2022.

[36] Guibiao Liao, Kaichen Zhou, Zhenyu Bao, Kanglin Liu, and Qing Li. Ov-nerf: Open-vocabulary neural radiance fields with vision and language foundation models for 3d semantic understanding. *IEEE Transactions on Circuits and Systems for Video Technology*, 2024.

[37] Derek Lim, Haggai Maron, Marc T Law, Jonathan Lorraine, and James Lucas. Graph metanetworks for processing diverse neural architectures. *arXiv preprint arXiv:2312.04501*, 2023.

[38] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[39] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *Advances in Neural Information Processing Systems*, pages 34892–34916. Curran Associates, Inc., 2023.

[40] Kunhao Liu, Fangneng Zhan, Jiahui Zhang, Muyu Xu, Yingchen Yu, Abdulmotaleb El Saddik, Christian Theobalt, Eric Xing, and Shijian Lu. Weakly supervised 3d open-vocabulary segmentation. *Advances in Neural Information Processing Systems*, 36:53433–53456, 2023.

[41] Ruyang Liu, Chen Li, Yixiao Ge, Thomas H Li, Ying Shan, and Ge Li. Bt-adapter: Video conversation is feasible without video instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13658–13667, 2024.

[42] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[43] Fan Ma, Xiaojie Jin, Heng Wang, Yuchen Xian, Jiashi Feng, and Yi Yang. 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*, pages 13151–13160, 2024.

[44] Muhammad Maaz, Hanoona Rasheed, Salman Khan, and Fahad Shahbaz Khan. Video-chatgpt: Towards detailed video understanding via large vision and language models. *arXiv preprint arXiv:2306.05424*, 2023.

[45] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65 (1):99–106, 2021.

[46] Aviv Navon, Aviv Shamsian, Idan Achituve, Ethan Fetaya, Gal Chechik, and Haggai Maron. Equivariant architectures for learning in deep weight spaces. In *International Conference on Machine Learning*, pages 25790–25816. PMLR, 2023.

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

[48] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022.

[49] Zekun Qi, Runpei Dong, Shaochen Zhang, Haoran Geng, Chunrui Han, Zheng Ge, Li Yi, and Kaisheng Ma. Shapellm: Universal 3d object understanding for embodied interaction. In *European Conference on Computer Vision*, pages 214–238. Springer, 2024.

[50] Zhangyang Qi, Ye Fang, Zeyi Sun, Xiaoyang Wu, Tong Wu, Jiaqi Wang, Dahua Lin, and Hengshuang Zhao. Gpt4point: A unified framework for point-language understanding and generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26417–26427, 2024.

[51] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[52] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

[53] Pierluigi Zama Ramirez, Luca De Luigi, Daniele Sirocchi, Adriano Cardace, Riccardo Spezialetti, Francesco Ballerini, Samuele Salti, and Luigi Di Stefano. Deep learning on object-centric 3d neural fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[54] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.

[55] Hyeonseop Song, Seokhun Choi, Hoseok Do, Chul Lee, and Taehyeong Kim. Blending-nerf: Text-driven localized editing in neural radiance fields. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 14383–14393, 2023.

[56] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[57] Can Wang, Menglei Chai, Mingming He, Dongdong Chen, and Jing Liao. Clip-nerf: Text-and-image driven manipulation of neural radiance fields. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3835–3844, 2022.

[58] Can Wang, Ruixiang Jiang, Menglei Chai, Mingming He, Dongdong Chen, and Jing Liao. Nerf-art: Text-driven neural radiance fields stylization. *IEEE Transactions on Visualization and Computer Graphics*, 2023.

[59] Guangming Wang, Lei Pan, Songyou Peng, Shaohui Liu, Chenfeng Xu, Yanzi Miao, Wei Zhan, Masayoshi Tomizuka, Marc Pollefeys, and Hesheng Wang. Nerf in robotics: A survey. *arXiv preprint arXiv:2405.01333*, 2024.

[60] Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. *Advances in Neural Information Processing Systems*, 36:8406–8441, 2023.

[61] Tong Wu, Jiarui Zhang, Xiao Fu, Yuxin Wang, Jiawei Ren, Liang Pan, Wayne Wu, Lei Yang, Jiaqi Wang, Chen Qian, 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*, pages 803–814, 2023.

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

[63] Jiabo Ye, Haiyang Xu, Haowei Liu, Anwen Hu, Ming Yan, Qi Qian, Ji Zhang, Fei Huang, and Jingren Zhou. mplug-owl3: Towards long image-sequence understanding in multi-modal large language models. *arXiv preprint arXiv:2408.04840*, 2024.

[64] Xumin Yu, Lulu Tang, Yongming Rao, Tiejun Huang, Jie Zhou, and Jiwen Lu. 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*, pages 19313–19322, 2022.

[65] Dong Zhang, Shimin Li, Xin Zhang, Jun Zhan, Pengyu Wang, Yaqian Zhou, and Xipeng Qiu. Speechgpt: Empowering large language models with intrinsic cross-modal conversational abilities. *arXiv preprint arXiv:2305.11000*, 2023.

[66] Hang Zhang, Xin Li, and Lidong Bing. Video-llama: An instruction-tuned audio-visual language model for video understanding. *arXiv preprint arXiv:2306.02858*, 2023.

[67] Jingbo Zhang, Xiaoyu Li, Ziyu Wan, Can Wang, and Jing Liao. Text2nerf: Text-driven 3d scene generation with neural radiance fields. *IEEE Transactions on Visualization and Computer Graphics*, 30(12):7749–7762, 2024.

[68] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

[69] Allan Zhou, Kaien Yang, Kaylee Burns, Adriano Cardace, Yiding Jiang, Samuel Sokota, J Zico Kolter, and Chelsea Finn. Permutation equivariant neural functionals. *Advances in neural information processing systems*, 36:24966–24992, 2023.

[70] Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. Minigpt-4: Enhancing vision-language understanding with advanced large language models. *arXiv preprint arXiv:2304.10592*, 2023.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: the claims made in the article are also highlighted in the abstract and introduction.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: the limitations of the proposed approach are detailed in section 7.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: the article does not introduce new theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: we provide all the information related to the training protocol used in our experiments and the implementation details in Section 3, Section 4, and Section 5. Moreover, we provide additional details in the appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: all our experiments have been conducted on publicly available data, i.e., ShapeNeRF-Text, HST and ObjaNeRF-Text. Our newly introduced dataset, Spatial ObjaNeRF, the source code and the weights for all our models will be publicly released in case of acceptance.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: all the training and test details are reported in Section 3, Section 4, and Section 5.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: we did not conduct multiple trials for each model training and evaluation due to the large computational requirements needed to fine-tune the LLMs employed in our approach.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: In Section 3 and Section 4, we provide all the details about the computational resources of our experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: we reviewed the guidelines listed in the NeurIPS Code of Ethics and we confirm that our approach does not violate them.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: we do not foresee any direct path to using our solution for negative applications as it pertains describing digital twins of single objects.

    Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: the large language models employed in our approach, i.e., LLaMA2-7b and LLaMA2-13b, are already equipped with safeguards mechanisms, and our finetuning does not affect such features. Indeed, the datasets we used for training are not scraped from the web. They contain only textual annotations on safe contexts curated by humans.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: In the Appendix, we provide details about the licenses for: *(i)* the large language models used in our approach, *(ii)* the employed open source codebases and *(iii)* the datasets used in all our experiments.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.

- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: our Spatial ObjaNeRF dataset will be made publicly available, in case of acceptance, together with the documentation required for reproducing the experiments. Moreover, in case of acceptance, we will also release the source code of our solution.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: the paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: the paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: the development of our core method did not involve LLMs.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.

Appendix

A Bilinear Interpolation of tri-plane Features

In this Section, we describe the algorithm used to obtain feature vectors from our tri-plane representation, introduced in Section 3.

Inputs:

- Tri-plane feature maps: three orthogonal 2D feature planes (F_{xy}, F_{xz}, F_{yz}) of dimension $\mathbb{R}^{H\times W\times C}$.
- A 3D point $p = (x, y, z) \in \mathbb{R}^3$.
- Axis-aligned bounding box for coordinate normalization: $aabb_{\min}, aabb_{\max} \in \mathbb{R}^3$.

Output: Interpolated feature vector at point p, $f \in \mathbb{R}^C$.

A.1 Coordinate Normalization

The 3D point p is normalized to the unit cube $[0, 1]^3$:

$$coords_{01} = \frac{p - aabb_{\min}}{aabb_{\max} - aabb_{\min}} \in [0, 1] \tag{3}$$

A.2 Projection to 2D Triplane Coordinates

The normalized 3D point is projected onto the three feature planes:

$$p_{xy} = (u_{xy}, v_{xy}) = (coords_{01}.x, coords_{01}.y), \tag{4}$$
$$p_{xz} = (u_{xz}, v_{xz}) = (coords_{01}.x, coords_{01}.z), \tag{5}$$
$$p_{yz} = (u_{yz}, v_{yz}) = (coords_{01}.y, coords_{01}.z). \tag{6}$$

A.3 Bilinear Interpolation on Each Plane

Given a feature plane F_{xy} and the projected coordinate p_{xy}:

Step 1: Map to image grid coordinates

$$u_{\text{img}} = u_{xy}(W - 1), \quad v_{\text{img}} = v_{xy}(H - 1) \tag{7}$$

Step 2: Compute integer and fractional parts

$$i = \lfloor u_{\text{img}} \rfloor, \quad j = \lfloor v_{\text{img}} \rfloor \tag{8}$$
$$\delta_u = u_{\text{img}} - i, \quad \delta_v = v_{\text{img}} - j \tag{9}$$

Step 3: Retrieve four neighboring features

$$f_{00} = F_{xy}[j, i, :] \quad \text{(top-left)}$$
$$f_{10} = F_{xy}[j, i + 1, :] \quad \text{(top-right)}$$
$$f_{01} = F_{xy}[j + 1, i, :] \quad \text{(bottom-left)}$$
$$f_{11} = F_{xy}[j + 1, i + 1, :] \quad \text{(bottom-right)}$$

Step 4: Compute bilinear interpolation

$$f = (1 - \delta_u)(1 - \delta_v)f_{00} + \delta_u(1 - \delta_v)f_{10} + (1 - \delta_u)\delta_v f_{01} + \delta_u\delta_v f_{11} \tag{10}$$

Steps 1–4 are repeated for the other planes to obtain f_{xz} and f_{yz}.

A.4 Feature Combination

The final feature vector is obtained by summing the interpolated features from the three planes:

$$f = f_{xy} + f_{xz} + f_{yz} \in \mathbb{R}^C \tag{11}$$

B Additional implementation details

B.1 `weights2space`: spatially-aware NeRF weights tokenization

NeRF encoding. Each NeRF is implemented as an MLP composed of an input layer, $L = 3$ hidden layers with $H = 64$ neurons each, and an output layer. The input to the network is a I-dimensional vector, with $I = 75$, obtained by applying frequency encoding [45] to the input coordinates. The input layer has a weight matrix $\mathbf{W}_i \in \mathbb{R}^{75\times64}$ and bias vector $\mathbf{b}_i \in \mathbb{R}^{64}$. Each hidden layer uses weights $\mathbf{W}_l \in \mathbb{R}^{64\times64}$ and biases $\mathbf{b}_l \in \mathbb{R}^{64}$. The output layer, which predicts RGB values and density σ, has weight matrix $\mathbf{W}_o \in \mathbb{R}^{64\times4}$ and bias vector $\mathbf{b}_o \in \mathbb{R}^4$. The *weights2seq* module takes as input all weight matrices and bias vectors, stacked row-wise into a matrix $\mathbf{W} \in \mathbb{R}^{S\times H}$, where $H = 64$ and $S = I + L(H + 1) + H + 2 = 336$.

Tri-plane representation. The sequence of tokens $\mathbf{Y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N) \in \mathbb{R}^{N\times C}$, with $N = 3072$ and $C = 516$, produced by *seq2space*, is reshaped into a tri-plane feature representation $M = (F_{xy}, F_{zx}, F_{yz})$. Each plane F is a tensor of shape $\mathbb{R}^{H\times W\times C}$, where $H = W = 32$ and $C = 516$.

Decoder. The decoder $\mathbf{\Phi}$ 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. At each training step, for every camera pose, 55000 rays are cast, forming a training batch $\mathcal{R}$.

B.2 *Spatial LLaNA*: spatially-aware MLLM for NeRF

Training. *S-LLaNA*-7B and *S-LLaNA*-13B are trained on NVIDIA A100 GPUs following a two stages protocol as described in the main manuscript. The hyperparameters 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 leverage AdamW [42] 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 leverage AdamW [42] as optimizer, and a cosine learning rate scheduler.

Datasets. The full training set of *Spatial LLaNA* includes approximately 300K NeRFs with textual annotations. While the ObjaNeRF-Text training set [7] contains annotations across all categories —brief, detailed, and Q&A—its test sets (PointLLM test set and GPT4Point test set) include only manually annotated brief descriptions. Therefore, in the main paper, the brief captioning task is evaluated on both ShapeNeRF-Text and ObjaNeRF-Text, while the other tasks are evaluated solely on ShapeNeRF-Text. Spatial ObjaNeRF provides detailed spatial annotations for a subset of 100 NeRFs from the GPT4Point test set of ObjaNeRF-Text. The question paired to each ground-truth annotation of this dataset is: *"Provide a very detailed description of the object. Focus on the relative position of the parts of the object. Include as many details as possible."* This question does not belong to the training set of *S-LLaNA*.

C Ablation experiment on tri-plane resolution and token dimensionality

We conducted several experiments to analyze the sensitivity of *Spatial LLaNA* to the number of queries N (and consequently tri-plane resolution) and token dimensionality C. We report results in Table 7, where the **first row** refers to the official version of *S-LLaNA* presented in the main paper. The **second row** shows an experiment to analyze the impact of the number of queries, and thus of the

Table 7: **Ablation on tri-plane and token configurations.** Best results are in **bold**, runner-up is underlined. (SN-T: ShapeNeRF-Text, ON-T: ObjaNeRF-Text PointLLM test set)

			SN-T brief captioning		ON-T brief captioning		SN-T detailed captioning		SN-T Q&A		Spatial ObjaNeRF	
Model	**Tri-plane Res.**	**Token Dim.**	**S-BERT**	**SimCSE**	**S-BERT**	**SimCSE**	**S-BERT**	**SimCSE**	**S-BERT**	**SimCSE**	**S-BERT**	**SimCSE**
S-LLaNA-13b	32	516	78.98	79.98	45.44	46.18	80.98	82.90	83.60	84.17	78.25	80.58
S-LLaNA-13b	16	516	76.65	77.68	41.77	42.11	76.74	78.69	83.36	83.83	75.48	77.26
S-LLaNA-13b	32	252	76.77	76.33	42.25	42.36	76.47	78.07	83.27	83.49	76.62	78.43

tri-plane resolution. weights2seq and *S-LLaNA* were trained using N=768 queries. As each plane is obtained by reshaping the queries into 3 planes of spatial dimension $H \times W$, with $H = W$ in our experiments, the corresponding tri-plane resolution is $\sqrt{\frac{768}{3}} = 16$. Finally, the **third row** analyzes the sensitivity of our model to the token dimensionality C. *S-LLaNA* was trained with $N = 3072$ tokens (i.e., tri-plane resolution 32) as in the main paper, yet with a smaller token dimensionality $C = 252$.

Each column reports the S-BERT and SimCSE results on various tasks and datasets: ShapeNeRF-Text (SN-T) brief and detailed captioning and QA, ObjaNeRF-Text (ON-T) brief captioning, and Spatial ObjaNeRF.

We observe that reducing the tri-plane resolution or token dimensionality leads to a marginal performance drop (around 2% across all metrics). This behaviour suggests that models with higher capacity (i.e., higher resolution and dimensionality) can encode richer and more fine-grained features from the input NeRF. However, the relatively small drop also indicates that *S-LLaNA* remains robust and effective even with more compact and computationally efficient representations.

D Ablation experiment on multi-token NeRF-Language Models

In this section, we report a set of experiments designed to isolate the impact of the spatial information encoded within multi-token NeRF representations for NeRF-Language Models. Originally, LLaNA [7] leverages a single-token representation computed from the nf2vec encoder [53]. To investigate on the impact of our proposed spatially-aware NeRF encoding, we train LLaNA after removing the max-pooling layer at the end of this encoder, resulting in a multi-token representation. However, these tokens lack explicit spatial information, unlike those in *Spatial LLaNA*, where the spatial information is encoded within the NeRF tokens thanks to the specific training methodology of the `weights2space` meta-encoder. The purpose of this experiment is to determine whether the superior performance of *Spatial LLaNA* compared to LLaNA stems primarily from the multi-token representation or from the spatial information that is explicitly encoded within the NeRF tokens. The results of this experiment are reported in Table 8. Once again, *S-LLaNA* emerges as the best performing model. For LLaNA, the transition to multiple tokens yields minimal performance improvements and, in most tasks, actually degrades performance. The only exception is the detailed captioning task on ShapeNeRF-Text, where multiple tokens produce limited gains of approximately 2 points in Sentence-BERT and 0.20 in SimCSE metrics. A marginal 1-point improvement in Sentence-BERT is also observed for the Q&A task on ShapeNeRF-Text. Overall, these results suggest that employing multi-token representations for the multimodal input instead of single-token representations contributes negligibly to performance improvements in NeRF-Language tasks. In contrast, the spatially-aware tokens utilized by *S-LLaNA* prove crucial for achieving superior performance across the considered tasks. These findings strongly suggest that the ability to extract spatially-aware representations from the weights of the input NeRF is significantly more important than increasing the number of tokens in the representation.

Table 8: **Ablation experiment on multi-token representation** Best results are in **bold**, runner-up is underlined. (SN-T: ShapeNeRF-Text, ON-T: ObjaNeRF-Text PointLLM test set)

	SN-T brief captioning		ON-T brief captioning		SN-T detailed captioning		SN-T Q&A		Spatial ObjaNeRF	
Model	**S-BERT**	**SimCSE**	**S-BERT**	**SimCSE**	**S-BERT**	**SimCSE**	**S-BERT**	**SimCSE**	**S-BERT**	**SimCSE**
S-LLaNA-13b	**78.98**	**79.98**	**45.44**	**46.18**	**80.98**	**82.90**	**83.60**	**84.17**	**78.25**	**80.58**
LLaNA-13b	75.09	76.45	42.08	42.40	75.51	77.63	81.05	81.60	75.01	77.30
LLaNA-13b multi-token	74.06	74.84	36.31	38.62	77.02	77.83	82.04	81.51	74.69	76.88

Table 9: **Zero-shot NeRF classification task on ShapeNeRF-Text.** Best results are in **bold**.

Model	Input modality	Accuracy (%)
LLaNA-7b	NeRF	67.56
S-LLaNA-7b	NeRF	**68.67**
LLaNA-13b	NeRF	69.27
S-LLaNA-13b	NeRF	**71.85**

E Experiments on the general language understanding of *Spatial LLaNA*

While our primary focus is spatial reasoning, we have included downstream tasks that go beyond spatial understanding to assess whether *S-LLaNA* retains general reasoning capabilities. First, the brief captioning task on ShapeNeRF-Text, HST, and ObjaNeRF-Text involves generating concise textual descriptions focused on identifying the main subject of a scene. This task primarily evaluates global scene understanding, rather than spatial reasoning. As shown in the main paper, in this setting, *S-LLaNA* consistently outperforms LLaNA, demonstrating its ability to preserve a holistic understanding of the input NeRF.

Furthermore, we evaluated *S-LLaNA* on a zero-shot NeRF classification task on ShapeNeRF-Text following the protocol in [6], where the model must infer the object class based on its multimodal input. This task does not require fine-grained spatial reasoning. Table 9 provides results on this task.

These results confirm that *S-LLaNA* does not sacrifice holistic visual or linguistic understanding. Finally, the ablation study on LLM finetuning of Section 6 demonstrates that *S-LLaNA* performs robustly even without this step, achieving results comparable to the finetuned variant. Consequently, in this version of our framework, the pre-trained LLM remains untouched, and thus certainly preserves its original reasoning capabilities and world knowledge.

F Examples from Spatial ObjaNeRF

Figure 6 presents a comparison between the textual annotations from the test set of ObjaNeRF-Text [7] and those from Spatial ObjaNeRF. Specifically, since Spatial ObjaNeRF includes 100 annotated NeRFs from the GPT4Point test split of ObjaNeRF-Text, we provide examples from this subset. The annotations of Spatial ObjaNeRF clearly offer rich descriptions with spatial relationships between the objects in the scene. In contrast, the original ObjaNeRF-Text test set provides brief descriptions that focus primarily on the main subject of the scene. For completeness, we include a JSON file alongside this document that contains all textual annotations from Spatial ObjaNeRF, each paired with the corresponding `object_id` of the 3D model from the Objaverse dataset [18].

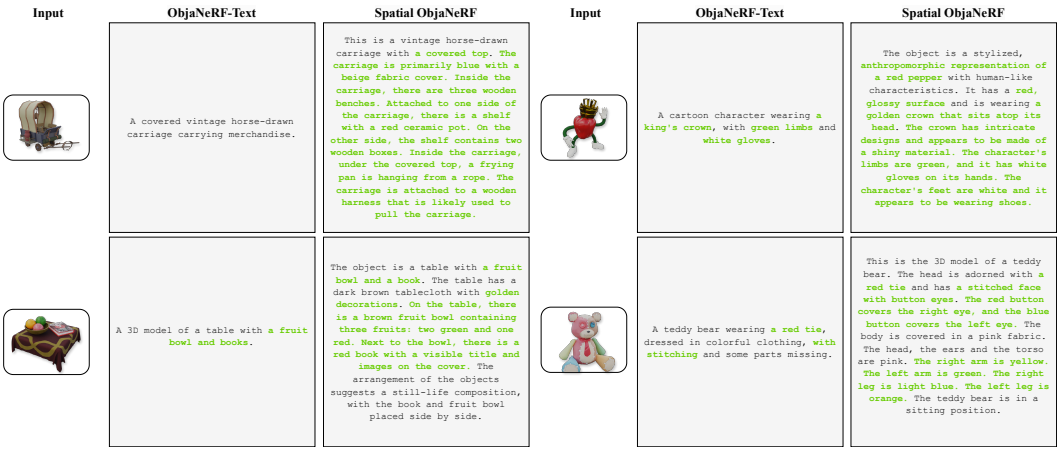


Figure 6: Comparison between ObjaNeRF-Text (GPT4Point test set) and Spatial ObjaNeRF annotations. In **green**, the fine-grained and spatial details.

G Real-world generalization of *Spatial LLaNA*

Although ObjaNeRF-Text already includes 3D models of real scenes, to further test the real-world generalization of *S-LLaNA*, we perform qualitative experiments on OmniObject3D [61] and mip-NeRF360 [10] datasets. The former provides meshes of real-scanned objects, captured in a controlled environment with neutral lighting and background, while the latter includes complex real-world unbounded scenes. mip-NeRF360 is one of the most commonly used datasets for novel view synthesis tasks. Figure 7 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 8, 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.

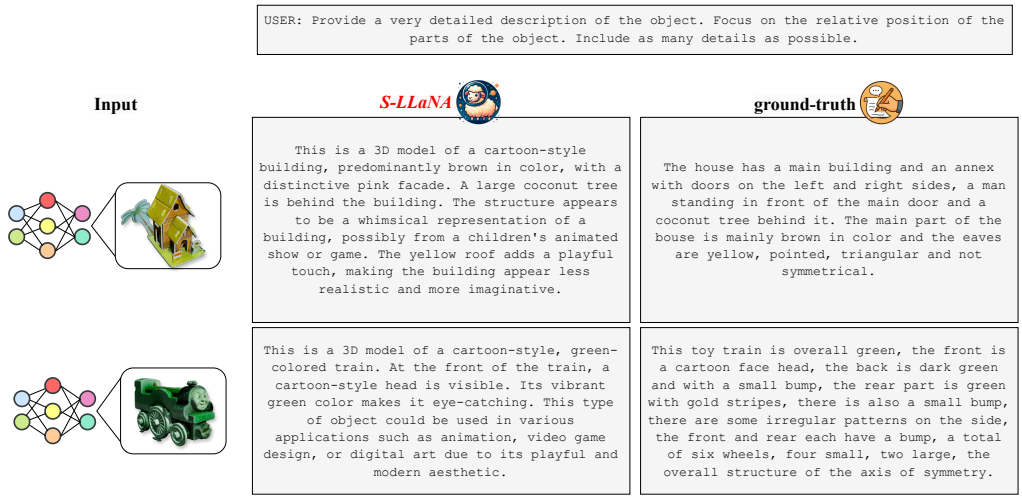


Figure 7: Qualitative examples of *S-LLaNA* on OmniObject3D

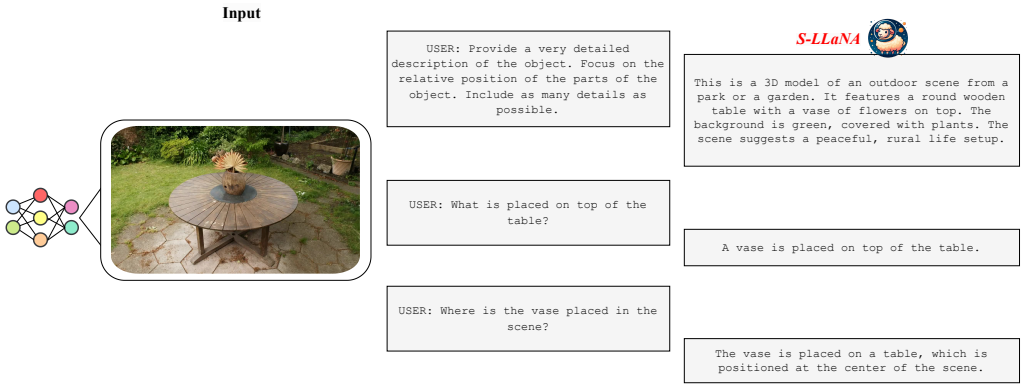


Figure 8: Qualitative examples of *S-LLaNA* on Mip-NeRF 360

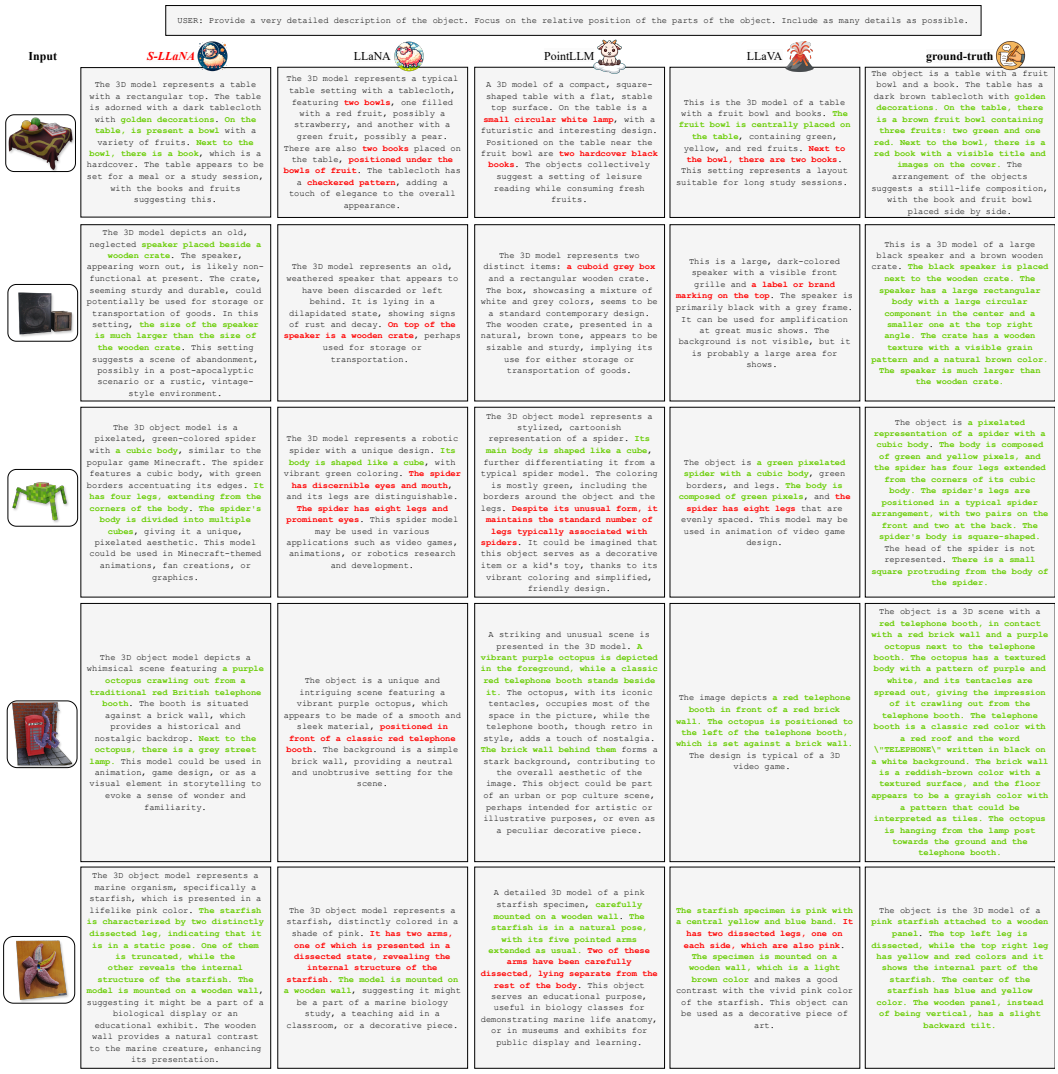


Figure 9: Qualitative evaluation on Spatial ObjaNeRF. In **red**, the hallucinations generated by the MLLMs. In **green**, the correct fine-grained and spatial details.

H Qualitative results on ShapeNeRF-Text, ObjaNeRF-Text and Spatial ObjaNeRF

Figure 9 shows qualitative results from the evaluation of the baselines on Spatial ObjaNeRF. This dataset contains a large set of scenes involving multiple objects, as shown in the Figure (first, second, and fourth row). It can be noticed how *Spatial LLaNA* (*S-LLaNA* in short) is capable of generating detailed and accurate descriptions of the input scenes, including fine-grained spatial information, without exhibiting signs of hallucination. In contrast, the baseline methods either omit such details or produce incorrect descriptions. The ground-truth annotation column further confirms that Spatial ObjaNeRF provides rich and comprehensive textual descriptions of the input 3D models. Figures 10 to 12 report qualitatives examples on the language tasks evaluated on the test sets of ShapeNeRF-Text and ObjaNeRF-Text [7]. In particular, since the test set of ObjaNeRF-Text contains only brief descriptions, this is the only task evaluated on this dataset, as in the experiments reported in the main paper. Results show that also on these tasks *Spatial LLaNA* is able to recognize and describe very specific details about the input scene, while the competing baselines are not able to do so.

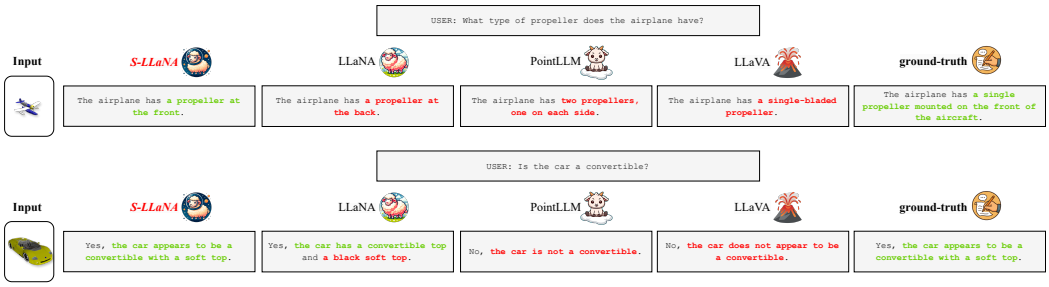


Figure 10: Qualitative evaluation of the single-round Q&A task on ShapeNeRF-Text. In **red**, the hallucinations generated by the MLLMs. In **green**, the correct fine-grained and spatial details.

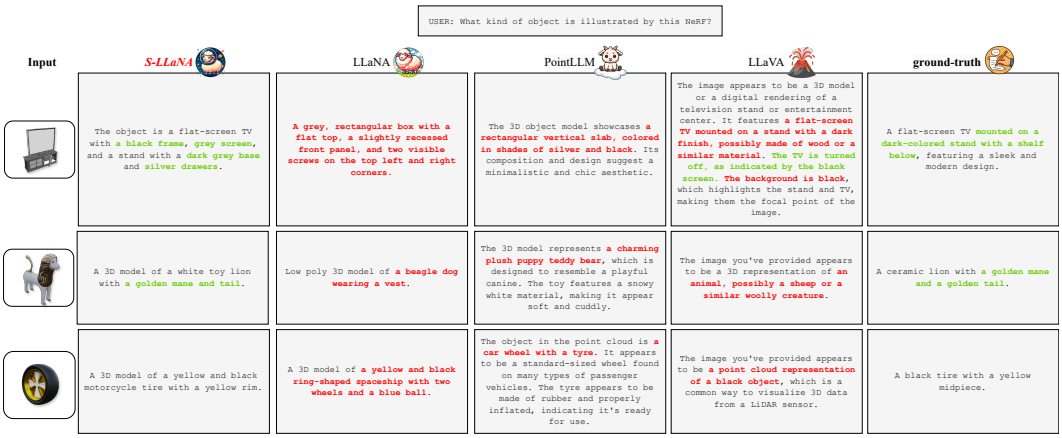


Figure 11: Qualitative evaluation of the brief captioning task on ShapeNeRF-Text and ObjaNeRF-Text. In **red**, the hallucinations generated by the MLLMs. In **green**, the correct fine-grained and spatial details.

I Failure cases of *Spatial LLaNA*

Figure 13 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 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.

J Additional visualization examples of the attention maps in LLaMA

In this section, we report additional visualization examples of the self-attention maps of LLaMA, the underlying LLM of *Spatial LLaNA*. While in the main manuscript we present the attention patterns from a single question word to the 3072 spatial NeRF tokens, here we show the opposite. More precisely, for a given NeRF token at a specific object location, we visualize the normalized attention scores to each text token in the question (Figure 14). 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

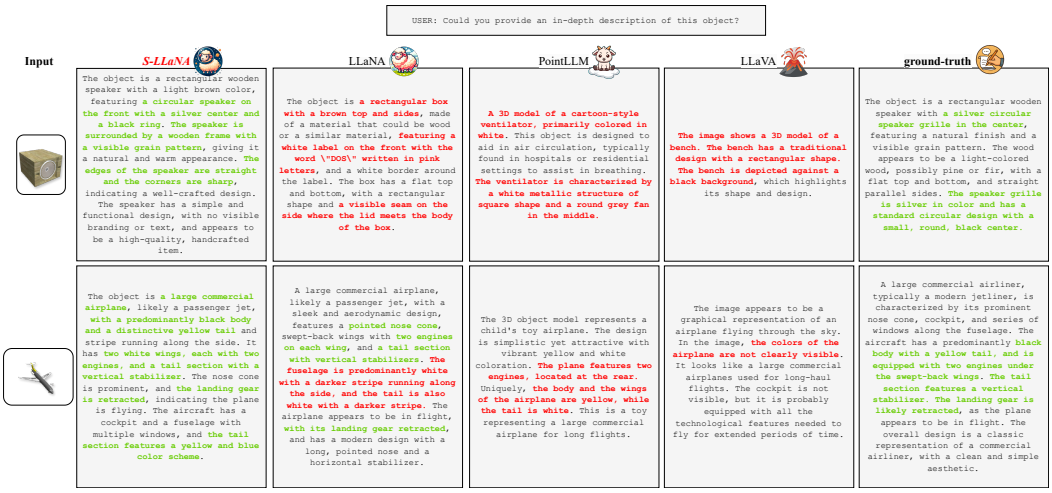


Figure 12: Qualitative evaluation of the detailed captioning task on ShapeNeRF-Text. In **red**, the hallucinations generated by the MLLMs. In **green**, the correct fine-grained and spatial details.

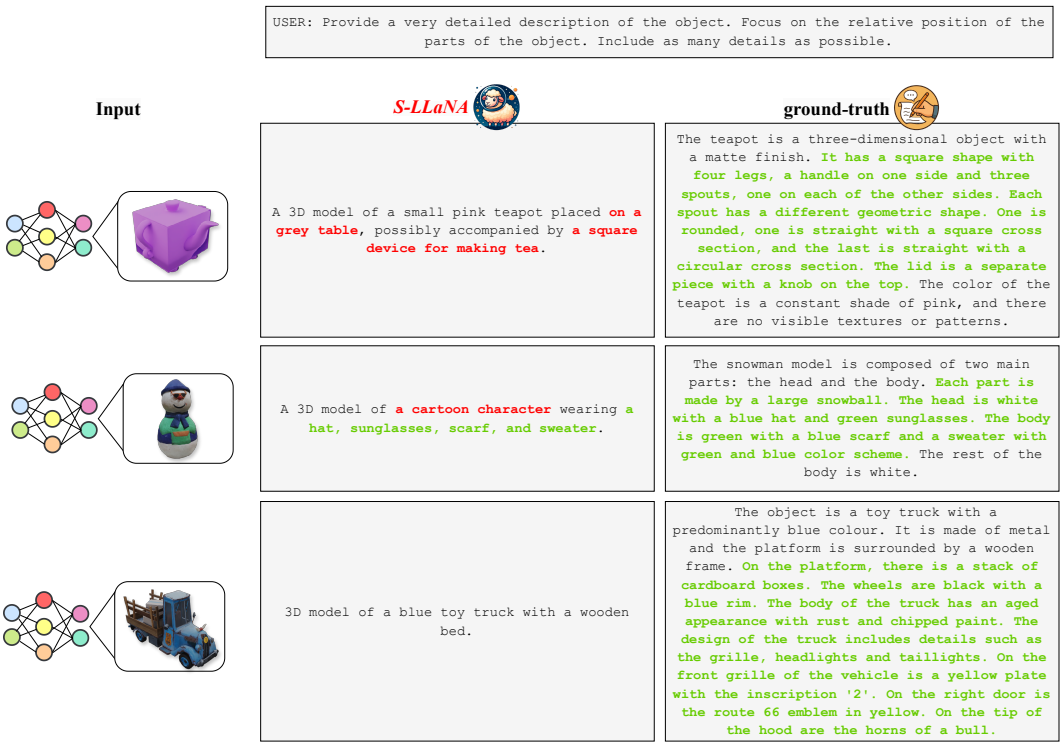


Figure 13: Examples of failure cases of *Spatial LLaNA* on Spatial ObjaNeRF. In **red**, the hallucinations generated by the model. In **green**, the correct fine-grained and spatial details.

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.

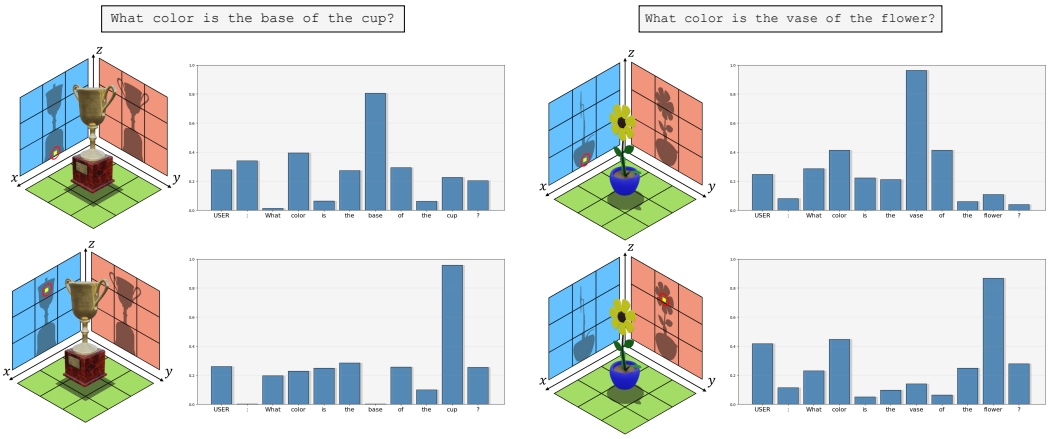


Figure 14: Visualization of the normalized self-attention scores of single NeRF tokens towards the text tokens of the question. In yellow, surrounded by a red circle, the reference NeRF token.

K Details about datasets, models and source code licenses

This section provides details about the datasets, models, and source code licenses used in the work, ensuring proper credit to the creators or original owners, and adherence to license terms.

Datasets: the datasets employed in our work and the relative licenses are listed below:

- **ShapeNeRF-Text**: licensed under MIT License.
- **ObjaNeRF-Text**: licensed under MIT License.
- **ShapeNet**: licensed under GNU Affero General Public License v3.0.
- **Objaverse**: licensed under Open Data Commons Attribution License v1.0 (ODC-By v1.0).
- **GPT2Shape HST**: licensed under Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.

Models: the models used in our experiments and their relative licenses are detailed below:

- **LLaNA**: licensed under MIT License.
- **nf2vec**: licensed under MIT License.
- **PointLLM**: licensed under Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.
- **GPT4Point**: licensed under Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.
- **ShapeLLM**: licensed under Apache License 2.0.
- **3D-LLM**: licensed under MIT License.
- **BLIP-2**: licensed under BSD 3-Clause License.
- **LLAVA**: licensed under Apache License 2.0.
- **LLAMA-2**: licensed under META LLAMA 2 COMMUNITY LICENSE AGREEMENT[1].

[1] https://ai.meta.com/llama/license/