Aligning Text, Images and 3D Structure Token-by-Token

Aadarsh Sahoo* Vansh Tibrewal* Georgia Gkioxari
California Institute of Technology

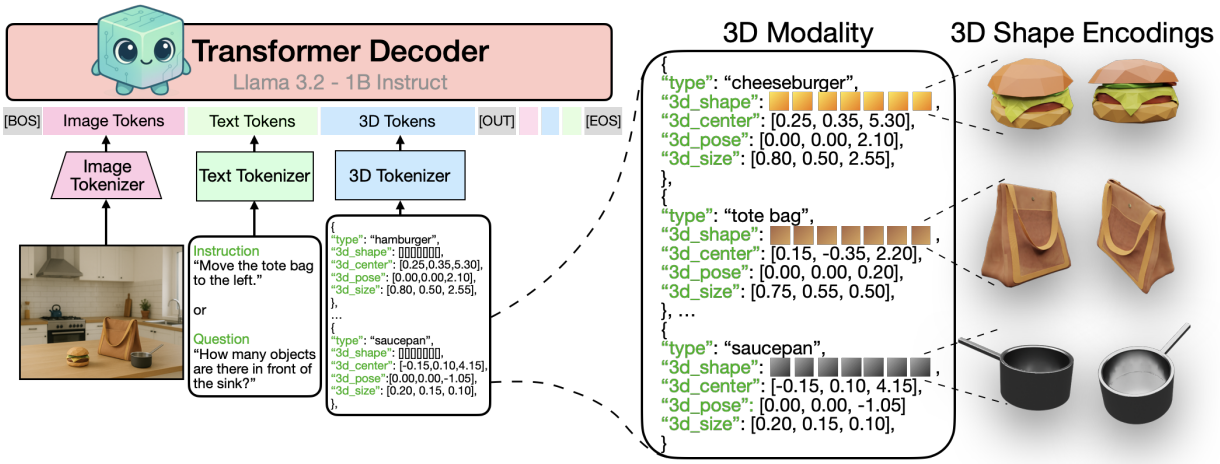


Figure 1. Kyvo: a decoder-only transformer aligns a structured 3D modality with language and vision. This 3D modality represents scenes as lists of objects, each defined by its 3D shape, type, 3D position, pose and size parameters. Kyvo unifies the token space of images, text, and 3D to enable a variety of complex visual 3D tasks.

Abstract

Creating machines capable of understanding the world in 3D is essential in assisting designers that build and edit 3D environments and robots navigating and interacting within a three-dimensional space. Inspired by advances in language and image modeling, we investigate the potential of autoregressive models for a new modality: structured 3D scenes. To this end, we propose a unified LLM framework that aligns language, images, and 3D scenes and provide a detailed "cookbook" outlining critical design choices for achieving optimal training and performance addressing key questions related to data representation, modality-specific objectives, and more. We show how to tokenize complex 3D objects to incorporate into our structured 3D scene modality. We evaluate performance across four core 3D tasks – rendering, recognition, instruction-following, and question-answering – and four 3D datasets, synthetic and real-world. We show our model's effectiveness on reconstructing complete 3D scenes consisting of complex objects from a single image and on real-world 3D object recognition tasks. Project webpage: https://glab-caltech.github.io/kyvo.

*Equal contribution

1. Introduction

Large language models (LLMs) that fuse text and images have unlocked unprecedented visual capabilities, such as visual captioning and text-guided image generation. Inspired by this success, we explore a third modality – structured 3D scenes – and show how aligning it with text and images allows LLMs to tackle a new suite of visual tasks in 3D, such as 3D reconstruction from a single image and 3D-conditioned image generation. We start from a language-pretrained transformer and extend it with a structured 3D modality that is designed to try to leverage its linguistic reasoning and generalization capabilities.

Our structured 3D modality encodes a scene as a list of its objects, where every object is specified by its 3D shape (*e.g.*, 3D mesh, 3DGS), type, 3D position, 3D size and 3D pose, shown in Fig. 1. This modality captures aspects of the physical world that are not directly conveyed through language or images alone. Additionally, it slots naturally into the unified token space shared by language and vision through an object-by-object tokenization scheme that integrates seamlessly with image and text tokens, so any modality can serve as input or output. As a result, it supports a broad range of tasks in 3D, such as image generation conditioned on the 3D scene structure (3D $\rightarrow$ image), predicting 3D objects, their shapes and locations from a single image (image $\rightarrow$ 3D),

and language-guided object-centric 3D editing (3D + image + text → 3D + image) – all within the same model design – shown in Fig. 2. These capabilities can reshape workflows. Robots can parse an image into a 3D scene composed of 3D objects, while designers can create complex scenes of objects through language in one forward pass of an LLM instead of wrestling with hard-to-use software like Blender.

But, how does one design and train an LLM that aligns this structured 3D modality with image and text? While there is extensive work on language-only [1, 14, 38] or vision-and-language models [6, 21, 39], there is limited work on training with an additional structured 3D modality. We explore the vast design space and evaluate the impact of design choices in architecture, training strategies, data representation, modality-specific objectives, and more. Our testbed consists of four core 3D tasks – image generation, recognition, instruction-following, and question-answering – and four challenging datasets, both synthetic and real-world. Through an extensive "cookbook" of rigorously validated empirical findings we hope to provide guidelines to design 3D-aligned multi-modal LLMs, including guidelines on tokenizing complex 3D shapes. We demonstrate the effectiveness of our unified LLM framework, Kyvo (Greek for "3D cube"), on single-view 3D reconstruction of complex object geometries. Finally, we apply our model to real-world domains of indoor and outdoor scenes for the task of image-based 3D object recognition, where we show that our model can tackle tasks that previously required task-specific vision specialists.

2. Related Work

LLMs. The success of LLMs is attributed to the scalability of transformers [42] taking the form of encoder-decoder [11, 36] and most popular decoder-only [1, 14, 38] models. LLMs showcase the effectiveness of autoregressive formulations in achieving task generalization, a finding we extend to 3D structured representations.

Vision-Language Models (VLMs). Early VLMs [3, 18, 32] learned from image–text pairs for tasks like zero-shot classification, retrieval, and QA. Modern VLMs [9, 23, 24, 28, 43, 45] extend these abilities using internet-scale data and stronger alignment. Recent studies [6, 21, 39] analyze vision-centric design choices but treat images only as input, while others [37, 40, 48] explore early fusion and diffusion for image generation. Nonetheless, current VLMs remain weak in 3D reasoning and geometry prediction, tasks that we tackle in this work.

LLMs & 3D. Prior work integrates LLMs with diverse 3D formats for tasks like VQA and grounding [27]. Some [10, 22, 26] extend VLMs to 3D via depth-aware queries, while others [16, 20, 29, 30, 33, 35, 47] embed 3D data (*e.g.*, point clouds, NeRFs) with CLIP-style features for open-ended QA. 3D-LLM [16] uses holistic 3D point clouds for captioning and QA, whereas our model's structured 3D modality decomposes scenes into objects, enabling direct alignment across language and vision. This supports tasks like 3D shape and pose prediction from a single image and image generation from object-centric 3D inputs. Similarly, while SceneScript [4] autoregressively predicts 3D boxes from videos and full scene clouds, our approach aligns images with structured 3D object and scene representations.

3D Tokenization. Prior work on 3D tokenization targets limited tasks, mainly single-asset generation. SAR3D [8] uses triplanes, while AToken [25] builds on Trellis [46] to train a joint tokenizer for images, 3D, and video. In contrast, we design a 3D tokenizer for structured scene encoding, optimized for compactness to represent multiple objects per scene. Whereas AToken and SAR3D use over 20k and 2040 tokens per asset respectively, our tokenizer achieves comparable or better reconstruction with far fewer tokens (see Sec. 3.2).

3. Building our model token-by-token

We present our approach, Kyvo, that aligns the 3D modality with text and images. The result is a unified multimodal LLM framework that can perform a range of visual 3D tasks – rendering, reconstruction, recognition and instruction-following – as shown in Fig. 2.

Sec. 3.1 outlines our experimental setup, including tasks and datasets. Sec. 3.2 presents a bottom-up analysis of key design choices – covering data representation, 3D tokenization, sequence design, and scaling – and addresses challenges in modality-specific tokenization and optimization. We summarize empirical insights and adopt optimal configurations for each subsequent experiment. Finally, we explore generalization to complex scene layouts (Sec. 3.3), 3D shape representations (Sec. 3.4), and real-world recognition (Sec. 3.5).

3.1. Setup: tasks and datasets

We design and train an autoregressive model aligning 3D with images and language, capable of performing 3D tasks.

Tasks. We focus on four core 3D tasks: image generation from 3D scene specifications (rendering), 3D object recognition & reconstruction from a single image, instruction-following 3D editing and question-answering.

Rendering: 3D → Image. Given a 3D scene described by our structured modality (object types, shapes, locations, poses), the model generates a corresponding image. This task tests whether rendering – typically requiring tools (*e.g.*, Blender) and complex processes (*e.g.*, ray tracing, rasterization) – can be reframed as feed-forward next-token prediction.

Reconstruction: Image → 3D. This is a dual of rendering. Given an image, the model predicts the underlying 3D scene structure, including object shapes, types, positions and poses.

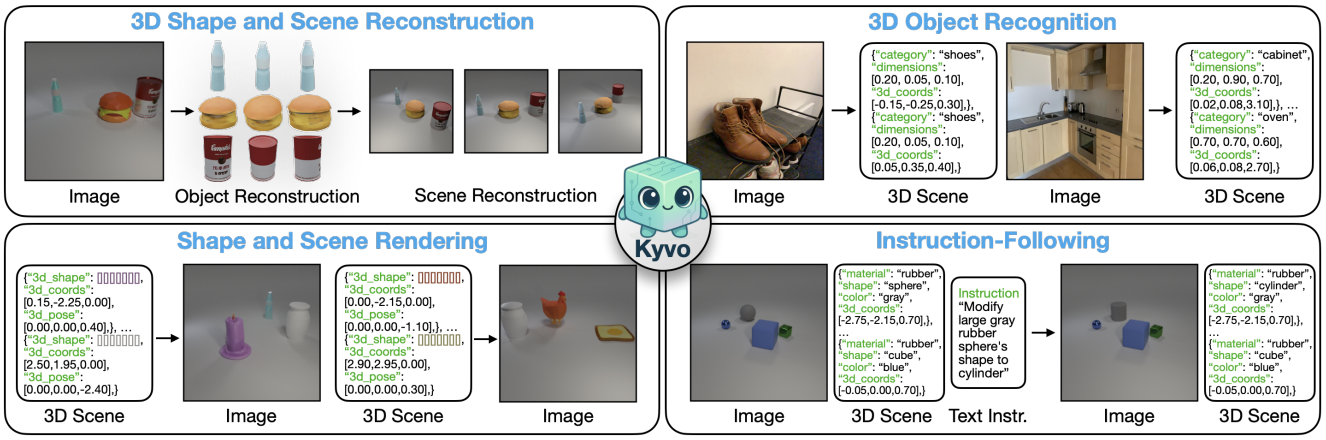


Figure 2. **3D task examples** with Kyvo's unified autoregressive framework using a structured 3D modality. (1) *3D shape and scene reconstruction:* From a single input image, Kyvo reconstructs individual objects with accurate geometry and spatial relationships. (2) *3D object recognition:* Given an input image, Kyvo identifies objects and predicts their 3D positions in real-world scenes. (3) *Shape and scene rendering:* Kyvo generates semantically consistent images from structured 3D scene inputs. (4) *Instruction-Following:* Given an image, 3D scene and text instruction, Kyvo produces coherent modifications to both image and the 3D representation.

Instruction-Following: $(\text{Image}, \text{3D}, \text{Text}_\text{I}) \rightarrow (\text{Image}, \text{3D})$. We define a set of tasks to manipulate and modify 3D scenes given a text instruction. These include: (1) *modifying the appearance of objects*, (2) *adding new objects*, (3) *removing objects*, and (4) *moving an object to a desired location*. For each subtask, we craft templated natural language instructions to evaluate the model's ability to follow instructions in 3D, *e.g.* "Remove the red mug behind the yellow bottle".

Question-Answering: $(\text{Image}, \text{3D}, \text{Text}_\text{Q}) \rightarrow \text{Text}_\text{A}$. We generate QA pairs using templated questions. Given the 3D scene, an image, and a query (*e.g.* "Is there a green object the same size as the glass vase?"), the model predicts a natural language answer. Details are in the Appendix.

All four tasks require spatial reasoning and grounding. Two of them explicitly involve text as input or output and all four implicitly require linguistic reasoning in the use of the structured 3D modality as input or output.

Datasets. We consider four datasets for our experiments. Two synthetic ones: CLEVR [19] features scenes of simple shapes in varying layouts; ObjaWorld features complex objects of any geometry sourced from Objaverse [12]. Two real-world ones: Objectron [2] and ARKitScenes [5] comprising real-world indoor and outdoor scenes of various object types.

Evaluation. Per our task definitions, Kyvo's output can be of either modality: text, image, or 3D.

Recognition. We evaluate predicted 3D scenes using the Jaccard Index, $J = \frac{tp}{tp+fp+fn}$, which measures object-level agreement between prediction and ground truth based on matching attributes (type, size, color, material) and spatial proximity within threshold τ. True positives are matched objects within τ, false positives are extra predictions, and false negatives are misses. We report the mean Jaccard Index over $\tau \in \{0.05, 0.10, 0.15, 0.20, 0.25\}$, reflecting both recognition and spatial accuracy.

Question-Answering. We report answer accuracy based on exact match between predicted and ground-truth text (typically 1–2 tokens). A random baseline yields 0.359 accuracy, while a frequency-based baseline reaches 0.430. See Appendix for more details.

Rendering. In the rendering task, models generate images from 3D scene inputs. Standard image metrics (L2, SSIM [44], FID [15], PSNR) fail to capture object placement or attribute accuracy. We therefore use human evaluations: annotators rank anonymized outputs against ground truth, and we report the mean rank. While L2 and SSIM roughly follow these trends, human judgment better reflects scene correctness (details in Appendix).

Instruction-Following. Here, the model predicts both images and 3D scenes. We report the Jaccard Index for 3D outputs, as this is the primary focus of our work.

3.2. Cookbook

Our model, Kyvo, starts from language as the principal modality. Our backbone is the decoder-only Llama-3.2-1B-Instruct [14] initialized from language-only pretrained weights. We extend it with modality-specific tokenizers for images and the structured 3D modality, along with modified input embeddings and output projections (Fig. 1). We do this in order to leverage the generalization and reasoning capabilities of LLMs, a hypothesis we later confirm in Tab. 5b. We evaluate key architectural choices – each with significant performance impact – which we hope offer insights to guide future multimodal LLM development in the 3D domain.

We begin by exploring how to represent and tokenize our structured 3D modality. We explore the tokenization of individual 3D objects of diverse geometry and appearance

Aux Rec Loss	Mean Rank(↓)
✗	2.828
Fixed Single View	1.672
Multiple Views	**1.500**

Table 1. Effect of auxiliary reconstruction loss.

3D Tokenizer	Mean Rank(↓)
SAR3D	1.605
Kyvo 3D VQ-VAE	**1.395**

Table 2. 3D tokenization comparison using human evaluation.

from Objaverse as well as the tokenization of scenes containing multiple objects of known geometry in CLEVR. This setup helps establish best practices for integrating 3D into a unified LLM framework. We then extend these findings to complex scenes in ObjaWorld (Sec. 3.3 & Sec. 3.4) and validate generalization to real-world scenes in Objectron and ARKitScenes (Sec. 3.5).

3.2.1. What is the optimal 3D representation?

Our model handles three modalities – images, text, and structured 3D scenes – by converting each into token sequences for autoregressive modeling.

Text. We employ an off-the-shelf text tokenizer from Llama-3.2 [14] with a 128,000 size vocabulary. Text instructions, questions, and answers are tokenized and enclosed within special, learnable tokens `[TEXT-START]` and `[TEXT-END]`.

Images. Our framework addresses tasks involving images as both inputs and outputs. To enable image generation within an autoregressive paradigm, we adopt discrete image representations using VQGAN [13]. This approach maps continuous image features to discrete tokens through a learned codebook. Specifically, we fine-tune a pre-trained VQGAN model to optimize the codebook for our visual domain. Image tokens are enclosed within special, learnable tokens, `[IMAGE-START]` and `[IMAGE-END]`.

3D scene tokenization. We represent 3D scenes as structured token sequences, where each list element encodes one object. Object attributes – shape, size, location, color, material – are expressed through special learnable tokens (e.g., `[SHAPE]`), whose values may be text ("car", "yellow"), numbers (for size or position), or learned 3D embeddings. Each object is enclosed by `[OBJECT-START]` and `[OBJECT-END]`, and the full scene by `[SCENE-START]` and `[SCENE-END]`. An example scene with two objects is shown below:

`[SCENE-START][OBJECT-START][SHAPE]<`v^1_1`,` v^1_2`,...,`v^1_{512}`>[LOCATION]-0.15 1.05 0.00 [POSE]0.00 0.00 3.00 [OBJECT-END][OBJECT-START][SHAPE]<`v^2_1`,`v^2_2`,...,`v^2_{512}`>[LOCATION]0.25 2.10 0.00[POSE]0.00 0.00 -2.05[OBJECT-END][SCENE-END]`

We discuss two key design decisions:

(1) How to tokenize 3D shapes (geometry & texture)? We aim for our structured 3D modality to encode complex objects via compact, autoregressively decodable 3D shape representations. To this end, we adopt Trellis [46], which

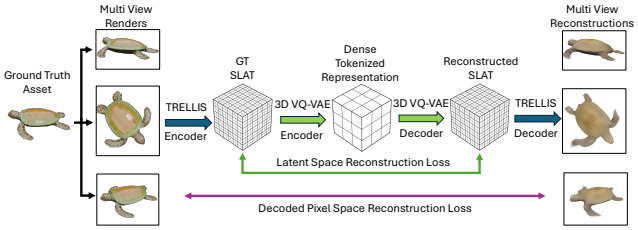


Figure 3. 3D VQ-VAE training involves the standard VQ-VAE losses (including reconstruction loss) applied in latent space as well as an auxiliary reconstruction loss applied in decoded pixel space.

encodes geometry and texture as sparse voxel features (slats) $z = \{(z_i, p_i)\}_{i=1}^{L}$, where each $z_i \in \mathbb{R}^8$ represents local features and p_i indexes active voxels in an N^3 grid. Although slats are sparse ($L \ll N^3$, typically $L \approx 20$k for $N = 64$), their length makes autoregressive modeling intractable.

We therefore train a 3D VQ-VAE [41] to compress slats from $64^3 \times 8$ to a dense $8^3 \times 128$ latent, vector-quantized with an 8192-token codebook. Each object is thus represented by 512 discrete tokens—a $\sim 40\times$ reduction—enabling efficient autoregressive decoding.

How do we train such a 3D VQ-VAE in latent slat space while preserving essential geometric and appearance information? We find that training the VQ-VAE in the latent space of slats is insufficient to learn an effective representation for reconstruction. To overcome this, we apply an auxiliary reconstruction loss to the decoded reconstructed slats in pixel-space, as shown in Fig. 3. We use the same pixel-space reconstruction loss ($\mathcal{L}_1$, D-SSIM and LPIPS) as in [46]. This yields significantly better reconstructions, despite achieving a similar latent space reconstruction loss, as shown in Fig. 4a. Additionally, we find that imposing a multi-view reconstruction loss of the asset (randomly sampled from 150 views) leads to better reconstructions than a single fixed view. We quantify the improvement using human evaluations in Tab. 1.

Our Trellis-based 3D VQ-VAE matches or surpasses SAR3D [8] while using 4× fewer tokens (512 vs. 2040). Qualitative and quantitive comparisons are shown in Fig. 4b and Tab. 2. Importantly, our compact 3D tokens integrate seamlessly into our unified vocabulary with image and language tokens, enabling efficient multi-object scene encoding and autoregressive decoding, while SAR3D tackles single-asset generation only.

While our representation enables effective 3D asset tokenization and reconstruction, we further test its suitability for autoregressive decoding. Using the same Llama-3.2-Instruct backbone, we evaluate reconstruction and rendering on unseen assets, in Fig. 4c. The learned 3D encodings generalize well, confirming their effectiveness for both reconstruction and decoding. These 3D tokens are then incorporated into our unified vocabulary alongside image and language tokens for training and inference.

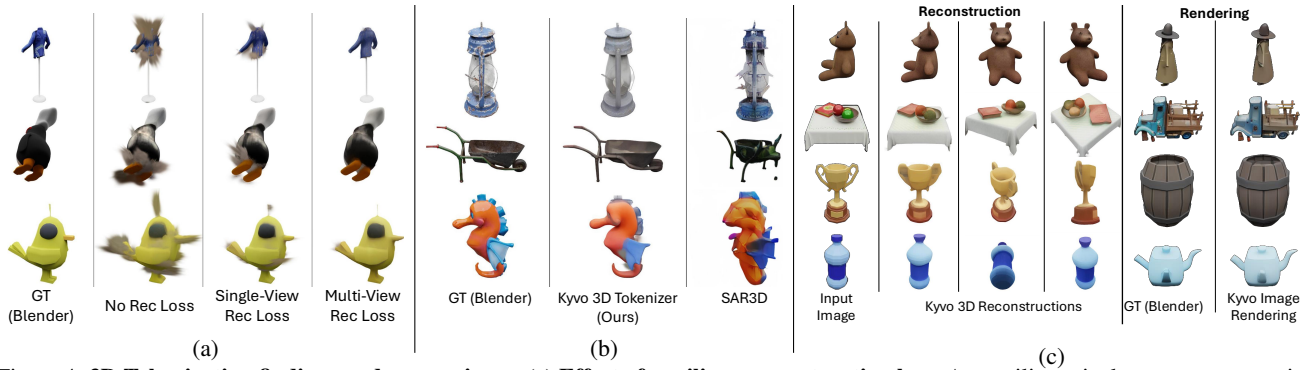


Figure 4. **3D Tokenization findings and comparisons.** (a) **Effect of auxiliary reconstruction loss.** An auxiliary pixel-space reconstruction loss on decoded renders from multiple views of the 3D object leads to much better reconstructions. (b) **3D tokenizer comparison.** Reconstructions from our Trellis-based VQ-VAE exceed the quality of SAR3D reconstructions with fewer tokens; improved textures stem from the Trellis slat representation rather than triplanes. (c) **Learned 3D shape encodings are effective during decoding.** The 3D tokens used in Kyvo are sufficient for both reconstruction and rendering using Llama 3.2 as decoder.

Granularity	Rendering(↓)	Recognition(↑)	Instruction(↑)	QA(↑)
0.005	1.380	0.5707	0.8643	**0.5185**
0.05	**1.200**	**0.9212**	**0.8666**	0.4980
0.5	2.020	0.2352	0.2427	0.4730

Table 3. **Effect of granularity.** A value of 0.05 yields the best overall performance on CLEVR.

(2) How to tokenize 3D location and orientation? In addition to 3D shape, the 3D location and orientation of an object are key attributes within our 3D modality and critical for understanding 3D spatial relationships and performing grounded 3D actions based on instructions. Thus, accurately encoding coordinates is essential. However, LLMs are struggle with numbers [31, 34]. To mitigate this, we encode each object's x, y, z coordinates separately as individual tokens, allowing the model to learn distinct embeddings for each coordinate. We discretize the coordinate values using equally spaced bins based on a chosen granularity. We find this granularity to be decisive for performance – if it is too coarse, spatial inaccuracies arise; if it is too fine, it leads to an exponential increase in tokens, with fewer training samples per bin and thus difficulty in learning. Tab. 3 shows the effect of granularity across the four tasks on CLEVR. We choose CLEVR, which features simple, known shapes, to isolate and study the effect of number encodings in next-token prediction frameworks. We find that a granularity of 0.05 outperforms the coarser 0.5 and the finer 0.005. Fig. 5 compares image generation for the rendering task on the test set across varying levels of granularity.

Furthermore, our tokenization approach substantially improves efficiency by reducing sequence length compared to standard text tokenizers, which typically fragment floating-point values (e.g., "0.000" becomes "0", ".", "000"). We achieve a mean sequence length reduction from 271.4 tokens using the standard Llama-3.2 tokenizer to 93.2 tokens – a 2.91× compression ratio. These efficiency gains directly translate to decreased memory requirements and faster training and inference times. The final vocabulary comprises 137,607 tokens covering all modalities.

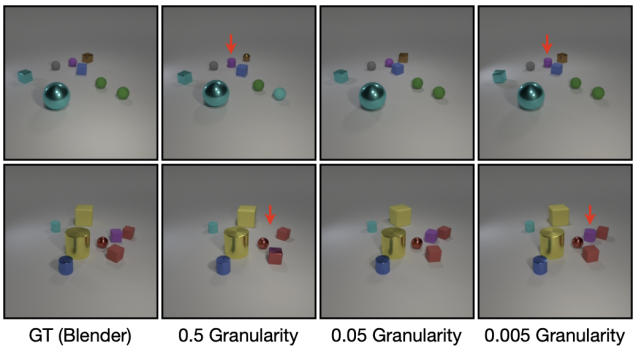


Figure 5. **Effect of Granularity.** A 0.05 granularity more accurately captures object locations and shapes.

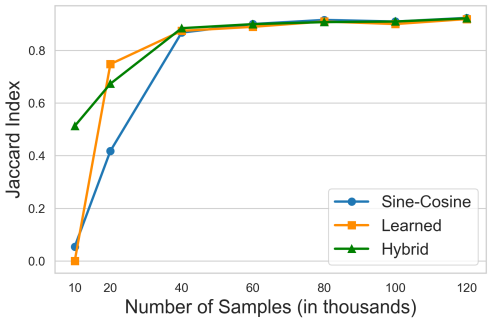


Figure 6. Hybrid number encodings are robust across data scales.

3.2.2. What matters in input sequence design?

We combine three modalities to construct the input sequence to the model. How should we order the modalities? And how should we encode numbers? We discuss our findings for input sequence design.

Sine-cosine encoding of numbers. While we independently tokenized coordinate values, naively learning embeddings for these tokens may fail to capture the inherent ordering of the numbers (*e.g.*, 2 is between 1 and 3). We investi-

CT reorder	Weighted loss	Rendering(↓)
✗	✗	2.66
✓	✗	3.56
✗	✓	2.78
✓	✓	**1.00**

Table 4. Effect of center-token reordering and weighted loss.

gate whether augmenting the learned embeddings with sine-cosine encodings can enforce numeric ordering relationships. Specifically, we evaluate three encoding strategies: (1) fixed sine-cosine encodings, (2) learned embeddings initialized from scratch, and (3) a hybrid approach where embeddings are learned but augmented with sine-cosine encodings. Fig. 6 plots the effect of these encoding strategies on the recognition task with varying training data sizes. While all methods perform on par in the high data regime, standalone fixed sine-cosine and learned embeddings collapse with low data. Consequently, we adopt the hybrid approach as it demonstrates robustness across data regimes. We show the performance of these encoding strategies across all four tasks in the Appendix.

Should the image or 3D come first? We investigate the impact of the ordering between the image and 3D modality for instruction-following and QA tasks. Our experiments reveal that placing the image before the 3D sequence leads to better performance compared to the reverse order. Specifically, we achieve an accuracy of 0.8666 with the sequence $(I, \text{3D}, T_I) \rightarrow (I, \text{3D})$ compared to 0.8350 with $(\text{3D}, I, T_I) \rightarrow (\text{3D}, I)$. Moreover, we obtain an accuracy of 0.4980 with $(I, \text{3D}, T_Q) \rightarrow T_A$ compared to 0.4720 with $(\text{3D}, I, T_Q) \rightarrow T_A$. This improvement could be attributed to the 3D tokens attending to the entirety of the preceding image tokens, enabling better conditioning and performance.

3.2.3. What matters in output sequence design?

We outline the important design decisions concerning the output sequence and objectives.

Initial token prediction matters. The next-token prediction scheme caused challenges in generating reliable image outputs during inference. Despite achieving low training loss, the model's decoded (next-token) predictions during inference deviated from expected outputs. At a high level, the issue stems from predicting rich outputs (images) from less informative conditions (3D specifications).

Further investigation revealed the issue: the first token. During inference, the model decodes sequentially, with the first token guiding the output. For CLEVR images, this token, representing the top-left corner, was biased toward a few codes – see Fig. 7a (blue dash plot) – due to CLEVR's uniform gray background. This caused overfitting and during inference a wrongly predicted first token caused the decoding to diverge. This finding is not just applicable to CLEVR but to any image set with a more uniform background, often the case in graphic design. Moreover, this trend is even evident with real-world images which we show in the Appendix.

To mitigate this issue, we incorporate a *center-token reordering* scheme to balance the token distribution at the sequence's starting position, by starting from the center token of the image and alternating hops after that as shown in Fig. 7a. This reordering means the first token now captures a representative part of the scene, instead of an uninformative background patch. The token distribution and method and results are shown in Fig. 7a and Fig. 7b respectively.

Token-specific loss weighting. We apply a *weighted loss* during training by assigning a higher weight (10.0) to the loss for the first five tokens of the output image sequence. This enforces a stronger constraint to correctly predict the initial tokens, which proved critical for autoregressive decoding.

Tab. 4 shows the impact of center-token reordering and weighted loss for the rendering task. Best performance is achieved when both are combined, also shown in Fig. 7b.

3.3. Generalization to complex scene layouts

Above, we validated our structured 3D modality and developed a cookbook for training unified autoregressive models with tokenized 3D shapes and attributes. We verified our findings for 3D shape encodings on diverse Objaverse objects and for 3D layout attributes (e.g., locations) on CLEVR scenes with known shapes but diverse layouts. We now show these findings generalize to ObjaWorld scenes with complex shapes (from Objaverse assets [12]) and diverse layouts.

We focus on two categories: park scenes (*person, bench, lamppost, bird*) and living room scenes (*person, sofa, coffee table*), featuring realistic textured objects with varied geometry at diverse locations and heights. We generate $50{,}000$ scenes per category (total $100{,}000$) to train rendering and recognition models, and $2{,}000$ additional scenes with unseen object layouts for testing. The rendering and reconstruction tasks are defined the same as with CLEVR.

For recognition, Kyvo achieves a Jaccard Index of 0.6415, well below its CLEVR score (0.9212, Tab. 3), reflecting ObjaWorld's increased complexity. Llama3.2-V, prompted with in-context examples for the same attributes, performs near zero, failing to accurately predict 3D locations.

For rendering, the model renders capture accurate object types, counts, and positions, though some errors occur in fine pose alignment (Fig. 8a).

Generalizing to novel scene configurations. We test on out-of-distribution inputs (*e.g.*, park-only objects placed in living room and vice-versa). Fig. 8b shows our model can generalize to such novel scenarios for the rendering task.

Chaining tasks. Chaining our recognition and rendering models enables image reconstruction to test model robustness. Fig. 8c shows an example of input image, chained-model reconstruction, and Blender rendering of the predicted scene. Blender shows accurate object types, poses, and positions, and the reconstructed images also closely match the

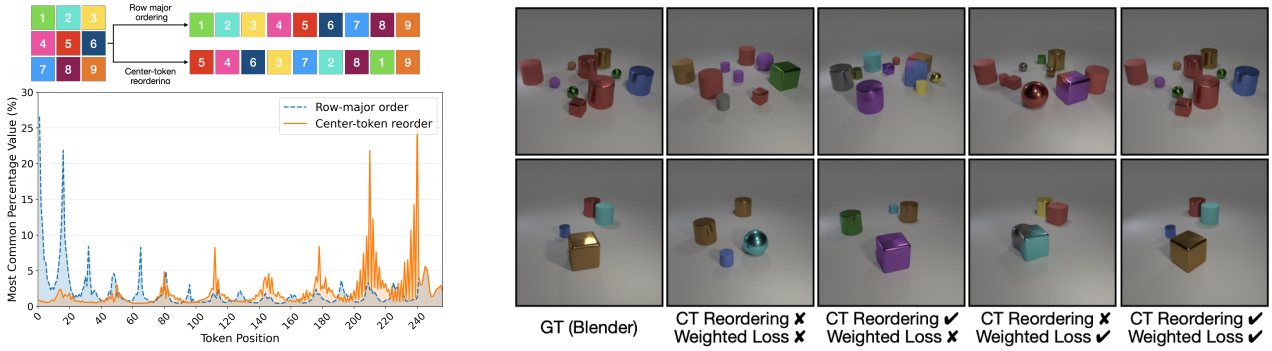


Figure 7. (a) **Top:** Schematic representation of center token reordering. **Bottom:** We convert CLEVR images into 256-token sequences and analyze token frequency. Over 25% of images share the same first token, causing biased predictions. Center token reordering significantly reduces this issue. (b) Inaccurate first-token predictions can cause catastrophic object- and scene-level diversions in autoregressive generations.

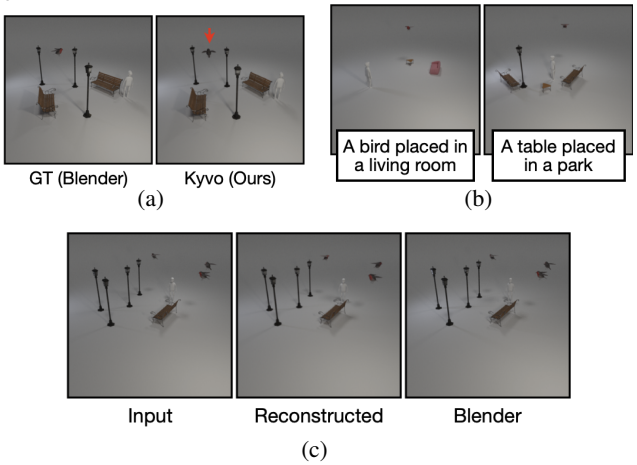


Figure 8. (a) **Rendering examples.** The model predicts images autoregressively from 3D inputs. Errors are largely pose mispredictions, *e.g.*, the bird. (b) **Novel scene configurations.** (c) **Chaining tasks.** Our recognition model predicts the 3D scene representation for an input image, which is then visualized through both our rendering model and Blender.

input, with only minor pose errors (*e.g.*, the birds).

3.4. Unified 3D shape and scene understanding

Sec. 3.3 focused on the effectiveness of our structured 3D modality for diverse scene layouts with *known* complex shapes. Therefore, we used type-specific object descriptions (*e.g.*, "bird" or "person"). Here, we bring together all our findings from Sec. 3.2, *including* the learned 3D shape tokenization. We show that Kyvo can reconstruct and render complex 3D objects and scenes, using a variant of ObjaWorld with complex objects (*e.g.*, barrel, chicken) from Objaverse [12]. We train recognition and rendering models on 100k image-scene pairs.

The recognition model must (1) reconstruct full 3D geometry for each object, and (2) infer each object's 3D position and pose from a *single image*. Fig. 9a shows results on unseen scenes: our model recovers object geometries and spatial layouts via our structured, object-centric 3D modality, while Trellis [46], a diffusion-based image-to-3D model trained on Objaverse that treats the scene holistically, often yields distorted shapes (*e.g.*, a deformed bottle) and misaligned layouts (*e.g.*, linearly arranged objects).

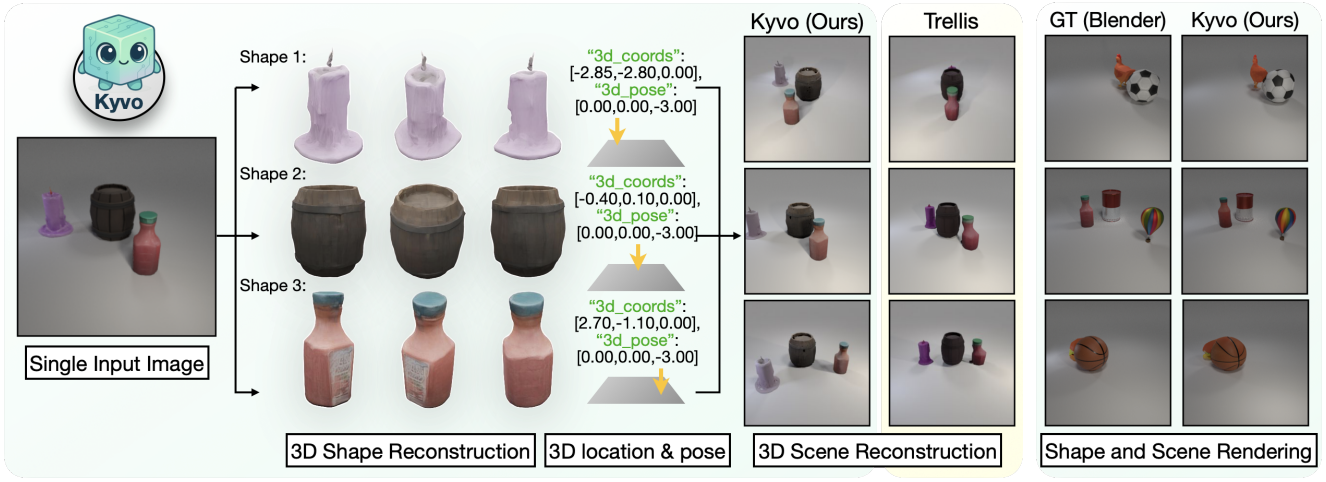
For rendering, the model takes the structured 3D modality with encoded shape tokens and outputs the corresponding image, mapping shape to appearance and placing objects at the correct positions and poses. Fig. 9b compares our outputs with Blender renderings, showing that our model reliably captures shapes and spatial relationships, with only minor distortions in challenging cases (*e.g.*, the occluded cheeseburger behind the basketball in the third image).

Qualitative scaling behavior. Fig. 10 shows that with limited data the rendering model produces amorphous color blobs that lack semantic coherence and captures only coarse layouts, while increasing training data progressively improves object geometry, appearance, and spatial relationships, yielding more accurate and consistent renderings.

3.5. Generalization to real-world recognition

We now evaluate our model's effectiveness for 3D object recognition in real-world scenes. We conduct experiments on two challenging real-world datasets: Objectron [2], which features indoor and outdoor scenes of diverse object categories (*e.g.*, bicycle, camera, car, cereal box); and ARKitScenes [5], featuring complex indoor environments of many object categories (*e.g.*, bathtub, bed, cabinet, chair). ARKitScenes presents additional complexity due to its scene density and ground truth annotation noise. Following the train-test splits by [7], we train a recognition model to detect objects by type and predict their 3D center coordinates and size dimensions in metric space. During training, we augment the data with random horizontal flipping.

Table 5a reports the Jaccard Index of our Kyvo and a state-of-the-art 3D object detector Cube R-CNN [7] – we apply a 0.05 confidence threshold to their predictions, as recom-



(a) (b)

Figure 9. (a) **Unified shape and scene reconstruction example.** Given a single input image, Kyvo predicts shape sequences and reconstructs individual objects (candle, barrel, bottle) along with their 3D locations and poses via our structured 3D modality, effectively reconstructing the 3D scene with consistent spatial relations between the objects, visualized using Blender. (b) **Shape and scene rendering examples.** Given the structured 3D modality as input, Kyvo renders images with consistent object appearance and spatial relationships.

Model	Objectron	ARKitScenes
Cube R-CNN (ResNet-34)	0.3276	0.2043
Cube R-CNN (DLA-34)	0.4012	0.2208
Kyvo (Ours)	0.4784	0.2118

(a) Jaccard Acc. vs Cube R-CNN.

Recipe	Rendering(↓)	Recognition(↑)	Instruction(↑)	QA(↑)
Scratch	1.36	0.6265	0.7744	0.4645
LoRA	1.82	0.8684	**0.8680**	0.3950
FFT	**1.26**	**0.9212**	0.8666	**0.4980**

(b) **Training recipe:** Scratch vs LoRA vs FFT.

Backbone	Rendering(↓)	Recognition(↑)	Instruction(↑)	QA(↑)
Llama-3.2-1B	1.38	0.8948	0.8674	0.4490
Llama-3.2-1B-Instruct	1.28	**0.9212**	0.8666	**0.4980**
Llama-3.2-3B-Instruct	**1.18**	0.8626	**0.8763**	0.2345

(c) **Effect of backbone** size and instruction-tuning.

Table 5. Comparing real-world recognition performance; Quantifying the effect of training recipes and different backbones for Kyvo.

Figure 10. Qualitative scaling behavior of rendering model.

mended by the authors. Kyvo significantly outperforms Cube R-CNN with two backbone variants on Objectron and performs on par on the more challenging ARKitScenes dataset. This result demonstrates the potential of a general autoregressive framework that aligns images with the 3D modality.

3.6. Analysis and observations

We explore the effects of training strategies and model backbones for Kyvo.

Training recipes. Table 5b compares three approaches for model adaptation: training from scratch, LoRA [17], and full fine-tuning (FFT). FFT from pre-trained language-only weights yields superior performance even when adapting to image and 3D modalities unseen during pre-training – suggesting effective cross-modal transfer with limited domain-specific data. Notably, LoRA performs worse despite its established efficacy for text-only adaptation, indicating limitations when incorporating entirely new modalities.

Instruction-tuned backbones and model sizes. Instruction-tuned backbones match or outperform non-instruction-tuned ones across all tasks, as shown in Table 5c. Increasing model size from 1B to 3B provides no significant gains (even performing notably worse for question-answering), indicating that the 1B model sufficiently captures our dataset's complexity while avoiding overfitting.

4. Conclusion & Limitations

We introduce Kyvo, a autoregressive model that aligns structured 3D with language and vision to support a broad range of 3D tasks. Our empirical "cookbook", based on training **307 models**, outlines effective design choices, including for tokenizing 3D scene attributes and complex 3D shapes. We will release code and data.

We cover limitations through scaling behavior, qualitative and quantitative results. A key challenge is the limited availability of 3D data. We show that strong performance and within-domain generalization can be achieved with relatively modest training data, but achieving cross-domain generalization demands larger datasets not readily available. A promising direction is extending Kyvo to handle mixed training data, enabling generalization to new domains even when 3D data is not always available as a paired modality.

Acknowledgments

We thank Damiano Marsili, Raphi Kang, Ilona Demler, and Ziqi Ma for their valuable feedback. Aadarsh is supported by the Kortschak Scholarship. Georgia is supported by the Powell Foundation, Meta through the LLM evaluation research grant, Google, and Amazon.

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

[2] Adel Ahmadyan, Liangkai Zhang, Artsiom Ablavatski, Jianing Wei, and Matthias Grundmann. Objectron: A large scale dataset of object-centric videos in the wild with pose annotations. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7822–7831, 2021. 3, 7

[3] 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. 2

[4] Armen Avetisyan, Christopher Xie, Henry Howard-Jenkins, Tsun-Yi Yang, Samir Aroudj, Suvam Patra, Fuyang Zhang, Duncan Frost, Luke Holland, Campbell Orme, et al. Scenescript: Reconstructing scenes with an autoregressive structured language model. In *ECCV*, 2024. 2

[5] Gilad Baruch, Zhuoyuan Chen, Afshin Dehghan, Tal Dimry, Yuri Feigin, Peter Fu, Thomas Gebauer, Brandon Joffe, Daniel Kurz, Arik Schwartz, and Elad Shulman. ARKitscenes - a diverse real-world dataset for 3d indoor scene understanding using mobile RGB-d data. In *NeurIPS Datasets and Benchmarks Track (Round 1)*, 2021. 3, 7

[6] Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, et al. Paligemma: A versatile 3b vlm for transfer. *arXiv preprint arXiv:2407.07726*, 2024. 2

[7] Garrick Brazil, Abhinav Kumar, Julian Straub, Nikhila Ravi, Justin Johnson, and Georgia Gkioxari. Omni3d: A large benchmark and model for 3d object detection in the wild. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13154–13164, 2023. 7

[8] Yongwei Chen, Yushi Lan, Shangchen Zhou, Tengfei Wang, and Xingang Pan. Sar3d: Autoregressive 3d object generation and understanding via multi-scale 3d vqvae. In *CVPR*, 2025. 2, 4

[9] Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, et al. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24185–24198, 2024. 2

[10] An-Chieh Cheng, Hongxu Yin, Yang Fu, Qiushan Guo, Ruihan Yang, Jan Kautz, Xiaolong Wang, and Sifei Liu. Spatialrgpt: Grounded spatial reasoning in vision language model. *arXiv preprint arXiv:2406.01584*, 2024. 2

[11] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53, 2024. 2

[12] 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. 3, 6, 7

[13] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12873–12883, 2021. 4

[14] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, and (additional authors not shown). The llama 3 herd of models, 2024. 2, 3, 4

[15] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017. 3

[16] 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*, 2023. 2

[17] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021. 8

[18] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International conference on machine learning*, pages 4904–4916. PMLR, 2021. 2

[19] Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2901–2910, 2017. 3

[20] Justin Kerr, Chung Min Kim, Ken Goldberg, Angjoo Kanazawa, and Matthew Tancik. Lerf: Language embedded radiance fields. In *ICCV*, 2023. 2

[21] Hugo Laurençon, Léo Tronchon, Matthieu Cord, and Victor Sanh. What matters when building vision-language models? *arXiv preprint arXiv:2405.02246*, 2024. 2

[22] Yuan-Hong Liao, Rafid Mahmood, Sanja Fidler, and David Acuna. Reasoning paths with reference objects elicit quantita-

tive spatial reasoning in large vision-language models. *arXiv preprint arXiv:2409.09788*, 2024. 2

[23] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36, 2024. 2

[24] Haoyu Lu, Wen Liu, Bo Zhang, Bingxuan Wang, Kai Dong, Bo Liu, Jingxiang Sun, Tongzheng Ren, Zhuoshu Li, Hao Yang, et al. Deepseek-vl: towards real-world vision-language understanding. *arXiv preprint arXiv:2403.05525*, 2024. 2

[25] Jiasen Lu, Liangchen Song, Mingze Xu, Byeongjoo Ahn, Yanjun Wang, Chen Chen, Afshin Dehghan, and Yinfei Yang. Atoken: A unified tokenizer for vision, 2025. 2

[26] Chenyang Ma, Kai Lu, Ta-Ying Cheng, Niki Trigoni, and Andrew Markham. Spatialpin: Enhancing spatial reasoning capabilities of vision-language models through prompting and interacting 3d priors. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. 2

[27] Xianzheng Ma, Yash Bhalgat, Brandon Smart, Shuai Chen, Xinghui Li, Jian Ding, Jindong Gu, Dave Zhenyu Chen, Songyou Peng, Jia-Wang Bian, et al. When llms step into the 3d world: A survey and meta-analysis of 3d tasks via multi-modal large language models. *arXiv preprint arXiv:2405.10255*, 2024. 2

[28] Yiyang Ma, Xingchao Liu, Xiaokang Chen, Wen Liu, Chengyue Wu, Zhiyu Wu, Zizheng Pan, Zhenda Xie, Haowei Zhang, Liang Zhao, et al. Janusflow: Harmonizing autoregression and rectified flow for unified multimodal understanding and generation. *arXiv preprint arXiv:2411.07975*, 2024. 2

[29] Ziqi Ma, Yisong Yue, and Georgia Gkioxari. Find any part in 3d. *arXiv preprint arXiv:2411.13550*, 2024. 2

[30] Damiano Marsili, Rohun Agrawal, Yisong Yue, and Georgia Gkioxari. Visual agentic ai for spatial reasoning with a dynamic api. In *CVPR*, 2025. 2

[31] Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. Gsm-symbolic: Understanding the limitations of mathematical reasoning in large language models. *arXiv preprint arXiv:2410.05229*, 2024. 5

[32] 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. 2

[33] William Shen, Ge Yang, Alan Yu, Jansen Wong, Leslie Pack Kaelbling, and Phillip Isola. Distilled feature fields enable few-shot language-guided manipulation. *arXiv preprint arXiv:2308.07931*, 2023. 2

[34] Aaditya K Singh and DJ Strouse. Tokenization counts: the impact of tokenization on arithmetic in frontier llms. *arXiv preprint arXiv:2402.14903*, 2024. 5

[35] Yuan Tang, Xu Han, Xianzhi Li, Qiao Yu, Yixue Hao, Long Hu, and Min Chen. Minigpt-3d: Efficiently aligning 3d point clouds with large language models using 2d priors. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 6617–6626, 2024. 2

[36] Yi Tay, Mostafa Dehghani, Vinh Q Tran, Xavier Garcia, Jason Wei, Xuezhi Wang, Hyung Won Chung, Siamak Shakeri, Dara Bahri, Tal Schuster, et al. Ul2: Unifying language learning paradigms. *arXiv preprint arXiv:2205.05131*, 2022. 2

[37] Chameleon Team. Chameleon: Mixed-modal early-fusion foundation models. *arXiv preprint arXiv:2405.09818*, 2024. 2

[38] Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023. 2

[39] Shengbang Tong, Ellis Brown, Penghao Wu, Sanghyun Woo, Manoj Middepogu, Sai Charitha Akula, Jihan Yang, Shusheng Yang, Adithya Iyer, Xichen Pan, et al. Cambrian-1: A fully open, vision-centric exploration of multimodal llms. *arXiv preprint arXiv:2406.16860*, 2024. 2

[40] Shengbang Tong, David Fan, Jiachen Zhu, Yunyang Xiong, Xinlei Chen, Koustuv Sinha, Michael Rabbat, Yann LeCun, Saining Xie, and Zhuang Liu. Metamorph: Multimodal understanding and generation via instruction tuning. *arXiv preprint arXiv:2412.14164*, 2024. 2

[41] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017. 4

[42] A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017. 2

[43] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024. 2

[44] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004. 3

[45] Zhiyu Wu, Xiaokang Chen, Zizheng Pan, Xingchao Liu, Wen Liu, Damai Dai, Huazuo Gao, Yiyang Ma, Chengyue Wu, Bingxuan Wang, et al. Deepseek-vl2: Mixture-of-experts vision-language models for advanced multimodal understanding. *arXiv preprint arXiv:2412.10302*, 2024. 2

[46] Jianfeng Xiang, Zelong Lv, Sicheng Xu, Yu Deng, Ruicheng Wang, Bowen Zhang, Dong Chen, Xin Tong, and Jiaolong Yang. Structured 3d latents for scalable and versatile 3d generation. In *CVPR*, 2025. 2, 4, 7

[47] Runsen Xu, Xiaolong Wang, Tai Wang, Yilun Chen, Jiangmiao Pang, and Dahua Lin. Pointllm: Empowering large language models to understand point clouds. In *ECCV*, 2024. 2

[48] Chunting Zhou, Lili Yu, Arun Babu, Kushal Tirumala, Michihiro Yasunaga, Leonid Shamis, Jacob Kahn, Xuezhe Ma, Luke Zettlemoyer, and Omer Levy. Transfusion: Predict the next token and diffuse images with one multi-modal model. *arXiv preprint arXiv:2408.11039*, 2024. 2