FORG3D: Flexible Object Rendering for Generating Vision-Language Spatial Reasoning Data from 3D Scenes

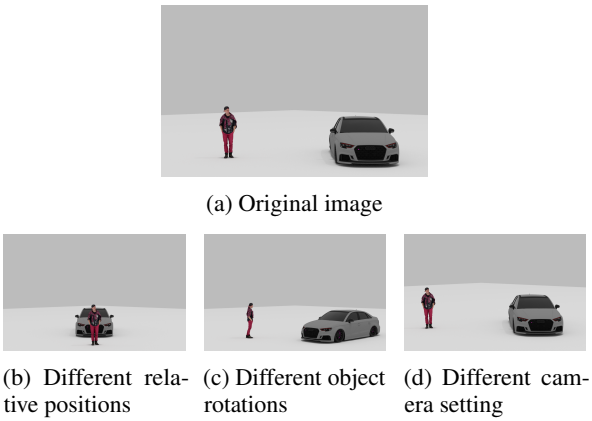
Oscar Pang[1,2,3] **Freda Shi**[1,2]
[1]: Vector Institute [2]: University of Waterloo [3]: University of Toronto
oscar.pang@mail.utoronto.ca fhs@uwaterloo.ca

Abstract

We introduce FORG3D, a 3D rendering toolkit developed with Blender and Python, which synthesizes vision-language data for two primary purposes: (1) supporting human cognitive experiments that require fine-grained control over material and (2) analyzing and improving the visual reasoning capabilities of large vision-language models. The toolkit provides flexible and precise control over object placement, orientation, inter-object distances, and camera configurations while automatically generating detailed spatial metadata. Additionally, it includes a built-in feature for integrating AI-generated backgrounds, enhancing the realism of synthetic scenes. FORG3D is publicly available at https://github.com/compling-wat/FORG3D, and a video demonstration is available at https://www.youtube.com/watch?v=QvIqib_PU8A.

1 Introduction

Spatial reasoning is a fundamental aspect of human cognition, where language is closely intertwined with visual perception to form a holistic understanding of the world (Landau and Jackendoff, 1993; Hayward and Tarr, 1995; Regier and Carlson, 2001; Levinson, 2003, *inter alia*). Cognitive scientists and psycholinguists have studied human spatial reasoning using diverse experimental materials, including text-only narratives (Bryant and Tversky, 1992; Bryant et al., 1992), 2D sketches or images (Carlson-Radvansky and Irwin, 1994; Logan, 1995), and simple 3D scenes (Li and Gleitman, 2002; Carlson and Van Deman, 2008; Bender et al., 2020) as the experimental material. However, developing 3D vision-language materials that simultaneously capture the complexity of real-world scenarios and maintain experimental control has remained a significant challenge due to the lack of easily accessible 3D rendering toolkits.



(a) Original image

(b) Different relative positions (c) Different object rotations (d) Different camera setting

Figure 1: Example rendered image showing a person facing to the right and a car facing the front and three rendered images of the same scene but with different configurations.

In machine learning, particularly the subfields of vision-language models (VLMs; Radford et al., 2021; Wang et al., 2024b; Liu et al., 2023b, *inter alia*) and embodied artificial intelligence (Li et al., 2024, *inter alia*), the ability to comprehend and reason about spatial relationships has become vital for applications such as image captioning, visual question answering, and robotic navigation. Despite their potential, current VLMs encounter challenges in spatial reasoning (Kamath et al., 2023; Liu et al., 2023a; Zhang et al., 2025), partially due to limitations in training data (Chen et al., 2024a; Ogezi and Shi, 2025)—existing datasets often lack spatial annotations and fail to adequately represent variations in object rotations, positions, and camera perspectives, thereby constraining the reasoning capabilities of VLMs.

Generating image-text pairs from 3D scenes holds the potential to address challenges in both cognitive experimental material design and vision-language model development. Along this line, we introduce FORG3D, a cross-platform 3D rendering toolkit developed using the Python interface of Blender 4.3 (Blender Team, 2024), specifically designed to generate high-quality vision-language

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations), pages 376–384

datasets for spatial reasoning tasks.

Functioning as a higher-level wrapper layer for the Blender rendering engine, FORG3D saves user effort in configuring complicated Blender environments and, thereby, empowers researchers with minimal Blender expertise to effortlessly create intricate 3D scenes by putting together objects on a planar surface. FORG3D uses objects under the Creative Commons license from Sketchfab,[1] and synthesizes diverse 3D scenes with high flexibility and controllability in object placement, orientation, and camera positioning (Figure 1), along with the accompanying metadata. This enables a pipeline to easily generate visual question answering or image captioning datasets based on custom 3D scenes, providing a comprehensive yet controlled environment that supports nuanced investigations of spatial reasoning. We anticipate that FORG3D will facilitate both cognitive science and multimodal machine learning research. The FORG3D toolkit is released under the MIT License.

2 Related Work

3D rendering toolkits and datasets for vision-language research. The most relevant work to ours is CLEVR (Johnson et al., 2017)—in addition to the widely used dataset, a data synthesis pipeline built with Blender 2.78 (Blender Team, 2016) has been released. The CLEVR synthesis pipeline allows researchers to generate synthetic data that controls color, size, and material (i.e., texture) for three simple objects, including cubes, spheres, and cylinders. Follow-up efforts have extended CLEVR for more complex visual reasoning tasks, such as referring expression comprehension (Liu et al., 2019) and physics understanding (Yi et al., 2020; Mao et al., 2022). Compared to them, FORG3D supports a wider range of objects, including but not limited to human figures, animals, vehicles, furniture, and buildings, and allows for more complex spatial configurations. Notably, the involvement of objects with an intrinsic frame of reference (FoR), such as humans, animals, and vehicles, enables the complex FoR-based analysis of spatial relations through rotations and translations of the objects (see Levinson, 2003, *inter alia*).

Synthetic datasets for training VLMs. Recent work has proposed to enhance the spatial reasoning abilities of VLMs using structured spatial priors (Cheng et al., 2024) or large-scale question-answer pairs (Chen et al., 2024b; Ogezi and Shi, 2025). However, the reliance on real-world photographs poses challenges in precisely interpreting spatial relations. Compared to them, FORG3D facilitates systematic diagnosis and potential improvement of large VLMs by providing precise 3D metadata alongside the rendered images.

Another line of work has proposed to incorporate 3D point clouds into VLMs (Hong et al., 2023, *inter alia*), which enriches the spatial perception of VLMs. However, the 3D point clouds are often resource intensive and require significant computational resources for training. In this work, we focus on generating 2D images from 3D scenes, which better aligns with the existing VLMs.

3D spatial reasoning benchmarks for VLMs. Several benchmarks have been introduced to evaluate and diagnose the spatial reasoning capabilities of VLMs, focusing on basic spatial relation recognition (Liu et al., 2023a; Kamath et al., 2023; Shiri et al., 2024; Wang et al., 2025), frame-of-reference adoption (Zhang et al., 2025), and cross-linguistic visual-question answering (Pfeiffer et al., 2022; Zhang et al., 2025). One major concern for these static benchmarks is the potential data leakage in training future models (Villalobos et al., 2024)—to this end, FORG3D supports dynamic benchmarking through generating unseen examples.

3 Methods

The FORG3D pipeline (Figure 2) supports controlling a broad range of factors in rendering scenes with two distinct objects on a planar surface. The scenes are annotated with precise spatial metadata. We offer support for and have tested extensively on Linux, Windows, and MacOS systems.

3.1 Framework

We formally define a scene, S, as a collection of the key parameters that generate it: the selected objects $(\mathcal{O}_1, \mathcal{O}_2)$, their relative spatial configuration $\mathcal{R}$, and the camera setup $\mathcal{C}$. The spatial configuration, $\mathcal{R}$, specifies the relative position of the second object to the first (e.g., 'left'), the individual rotations for each object (r_1, r_2), and the distance between them (d). The camera configuration, $\mathcal{C}$, contains values for tilt, pan, height, and focal length. The FORG3D pipeline then operates as a deterministic function, which we can denote as `Render(S)`, that maps this complete parameter set S to a pair of outputs: the rendered image, I, and corresponding metadata,

[1] https://sketchfab.com/

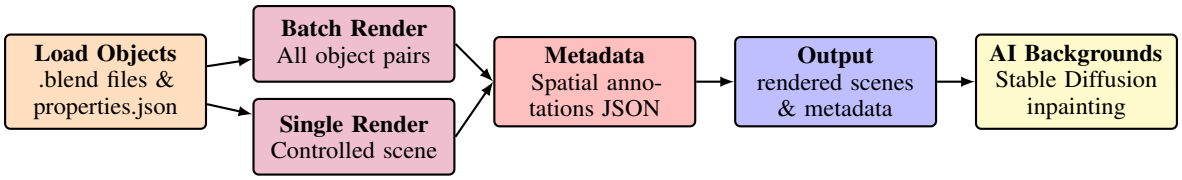


Figure 2: Compact pipeline diagram of the FORG3D tool. Objects are loaded, scenes are rendered in batch or single mode, metadata is generated, an output folder is created with the rendered scenes and metadata, and optional AI-generated backgrounds can be added to the output images.

M. This metadata is a direct record of the parameters in S, ensuring that every image is paired with its exact ground-truth data for reproducibility.

3.2 Rendering Pipeline Setup

The pipeline initializes by integrating the project repository with the Blender Python environment. This involves configuring Blender to recognize the FORG3D source directory through a dedicated `.pth` file placed in its `site-packages`. Since the rendering tool relies solely on libraries that are pre-installed in the Blender 4.3 Python environment, no additional dependencies are required. Users can either load our 21 preset objects from an external repository by running the provided shell script or add custom objects as `.blend` files and label their properties in `properties.json`.

3.3 Camera Configuration

FORG3D provides extensive customization of camera parameters, allowing for controlled manipulation of the viewpoint. The supported camera settings include:

- Tilt: vertical angle of the camera;
- Pan: horizontal angle of the camera;
- Height: camera's vertical position;
- Focal length: camera's zoom.

These parameters are supplied via command-line arguments or configuration files, facilitating reproducibility across experiments.

3.4 Object Spatial Configurations

The current version of FORG3D is designed to render scenes with two objects, with the potential to extend to more in the future. There are two primary rendering modes:

Batched rendering (`--render-random`). Under this mode, the toolkit automatically renders scenes for all pairs of objects found in a specified data directory. The output is organized into subdirectories labeled according to the relative positioning of the objects:

- `[object1]_[object2]_{left, right, front, behind}`

For each of these subdirectories, the system generates renderings that encompass all possible combinations of rotations around the vertical z-axis, where each rotation of `object1` is paired with each rotation of `object2`, capturing a full range of variations in orientations and relative perspectives between the two objects. If no camera configuration is specified, each of these renderings will be repeated a specified number of times, for each of the manually created configuration settings that include all combinations of tilt, pan, height, and focal length. These settings are created in the source code to ensure optimal visibility of the scene, as poorly chosen camera settings can obscure or distort the view, making it difficult to observe the objects clearly. The additional parameters that can be specified in the command line are listed as follows:

- Object selection;
- Distance between the two objects;
- Maximum number of images to render for each subdirectory (various rotations);
- Camera configurations.

Furthermore, FORG3D supports object overlap prevention to ensure clarity and object visibility. Specifically, we discard images where (1) a smaller object is hidden behind a larger one (determined by $> 75\%$ overlap of bounding box pixels) or (2) objects positioned side-by-side share common pixels.

Single-image rendering (default option). In this mode, additional parameters enable precise control over the spatial relations:

- Position of `object2` relative to `object1`;
- Individual object rotations around the z-axis in degrees (clockwise).

This dual-mode functionality allows for both exhaustive dataset generation and targeted experimental material synthesis.

3.5 Metadata Generation and Organization

After each image is rendered, a corresponding JSON metadata file is generated, containing detailed information on applied camera settings and object transformations. The metadata encapsulates:

- Numerical values for camera tilt, pan, height, and focal length.
- Positions (x-y coordinates) and orientations (left, right, front, behind) of the objects.
- Spatial relationship between the objects from the viewer's perspective, as well as the relative perspectives of both objects

This rigorous documentation of scene parameters is critical for reproducibility and systematic analyses.

3.6 AI-Generated Background Integration

As an extended feature, FORG3D can optionally integrate AI-generated background using the Stable Diffusion XL inpainting model (Rombach et al., 2022), which modifies the background pixels while preserving the objects. Every time an image is rendered, a corresponding masked image is also saved, with the background being white and the objects colored black. By executing the provided script with a custom prompt, users can mask out the default plain backgrounds of the rendered images and replace them with more realistic environments generated by the model. This ensures that the original objects and their spatial relationships remain intact while introducing diverse contextual settings. Users can also customize the diffusion model's parameters in the Python script, including `guidance_scale` (creativity), `num_inference_steps`, and `strength`. This feature works best on images with square resolutions, as the inpainting model is optimized for those dimensions.

3.7 Controlled and Uncontrolled Elements

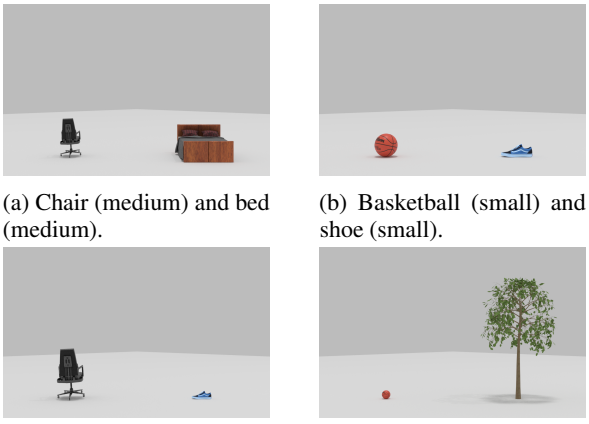
The rendering pipeline balances precision and flexibility through controlled, semi-controlled, and uncontrolled elements. Fully controlled elements include the objects themselves, object positions defined by relative relationships, object orientations, inter-object distances, object scaling (set in the `properties.json` file), image resolutions (set in the `config.json` file), and camera settings.

Semi-controlled elements are those that offer customization with limits. For instance, the backgrounds generated with Stable Diffusion allow users to replace plain defaults with realistic scenes using custom prompts, though the exact details of the backgrounds depend on the model. Object texture is another feature being semi-controlled, requiring manual application in Blender for material properties, outside the automated pipeline.

Uncontrolled elements are those that lie beyond direct manipulation, imposing limitations on customization. For example, scene lighting defaults to the uniform setup from Blender, with no control over directional sources of shadows. Additionally, the specific positions of the two objects in each scene cannot be set using coordinates. Instead, their positions are calculated in the source code, using the relative directions of the objects, for consistency. However, the implementation maintains sufficient flexibility to accommodate these additional controls in future developments.

4 Demonstration

In this section, we present examples generated by FORG3D. For detailed descriptions of each function's parameters and return values, refer to the official documentation.[2]



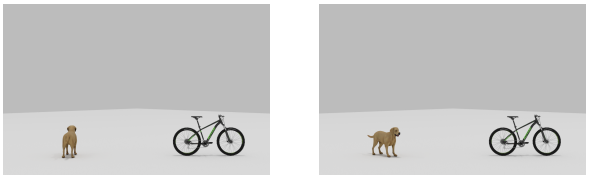
(a) Chair (medium) and bed (medium).

(b) Basketball (small) and shoe (small).

(c) Chair (medium) and shoe (small).

(d) Basketball (small) and tree (large).

Figure 3: Rendered scenes of various object pairs.

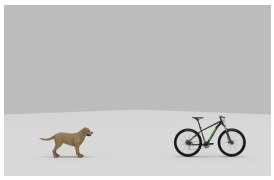


(a) Dog facing behind.

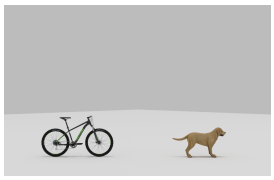
(b) Dog facing front-right (45 degrees).

Figure 4: Example rendered scenes of a dog and a bike with the dog facing different orientations.

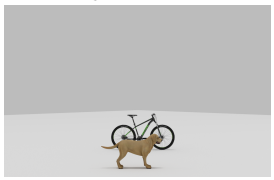
[2] https://compling-wat.github.io/FORG3D/



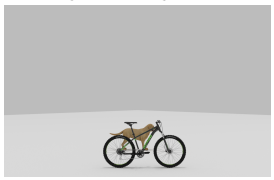
(a) Dog to left of bike.



(b) Dog to the right of bike.

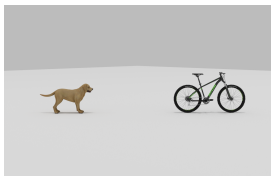


(c) Dog in front of bike.

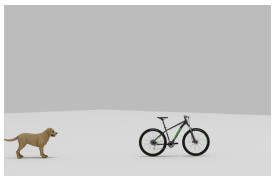


(d) Dog behind bike.

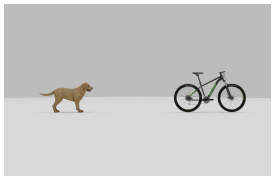
Figure 5: Example rendered scenes of a dog and a bike with different relative positions.



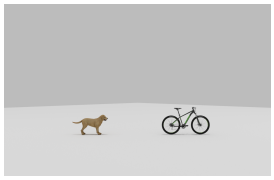
(a) Camera tilted down.



(b) Camera panned right.



(c) Camera shifted down.



(d) Camera zoomed out.

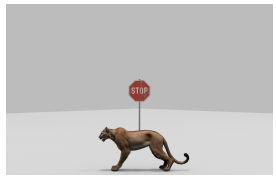
Figure 6: Various camera configurations for the scene in Figure 5a.

Various object combinations. Any pair of Blender objects can be rendered into a scene. For each pair of objects rendered, the objects are scaled according to their size groups recorded in `properties.json`. For example, the basketball, when placed next to the shoe, which is classified as a "small" object, appears larger than when it is placed next to the tree, which is classified a "large" object (Figure 3).

Different orientations. The toolkit supports rendering images with different orientations of the objects, which can be specified in the command line or configuration files (Figure 4).

Relative positions. The toolkit supports rendering images with different relative positions of the objects, which can be specified in the command line or configuration files (Figure 5). Data synthesized with respect to relative positions can be used to reproduce and validate the generalizability of results by Zhang et al. (2025).

Camera configurations. The toolkit supports rendering images with different camera configurations, which can be specified in the command line or configuration files (Figure 6). Data synthesized with respect to camera configurations can be used to study human and model preference towards certain linguistic descriptions of spatial relations from different angles of views, which, to the best of our knowledge, has not been studied in the literature.



```
{
  "camera": { },
  "ground_object": {
    "name": "puma",
    "orientation": "left",
    "intrinsic_caption":
        "From the puma's perspective,
        the sign is to the right of it."
  },
  "figure_object": {
    "name": "sign",
    "orientation": "front",
    "intrinsic_caption":
        "From the sign's perspective,
        the puma is in front of it."
  },
  "translational_relation_caption":
    "The sign is in front of the puma.",
  "reflectional_relation_caption":
    "The sign is behind the puma."
}
```

Figure 7: Example rendered image (top) and corresponding syntax-highlighted JSON metadata (bottom). The camera configuration is omitted for brevity.

(a) Original image of a hut and a tree.

(b) With AI-generated background.

Figure 8: Example of applying the background generation process to a rendered image.

Example metadata. The JSON metadata file for each rendered image is mostly self-explanatory (Figure 7), containing the camera configuration for the scene, both objects' rotations, positions, and orientations relative to the camera, as well as the scene captions. The `intrinsic_caption`

field for an object represents the description of the scene from the perspective of that object, while the last two fields specify the objects' spatial placements from the viewer's perspective in `translational` and `reflectional` contexts (Levinson, 2003), respectively—the former treats the direction towards the background as the front, while the latter treats the direction towards the camera as the front.

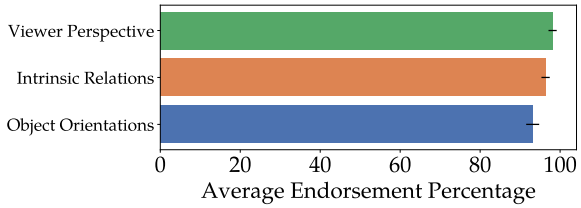
AI-generated backgrounds. Figure 8 presents an example of applying the inpainting model to generate a background based on the original rendered image, with the prompt: *"realistic sky and ground, textures, colours, lighting, detailed."*

5 Quantitative Evaluation

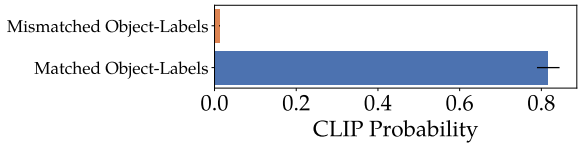
We generate a dataset of rendered images using FORG3D, which includes 210 unique pairs from 21 objects using the `render_multiple.sh` script. Each pair is rendered in four relative positions, organized into separate subdirectories, with at most five variations in object rotations per subdirectory and at most four camera configurations per scene. The process took approximately 5 hours of user time and 2 hours of system time on a machine equipped with an NVIDIA RTX 4090 GPU.

The metadata co-generated with the images have provided spatial orientation and positional details necessary for generating captions and corresponding questions. The format of the questions was taken from specific categories from the 3DSR-Bench benchmark (Ma et al., 2024), which is a dataset of multiple-choice questions related to the relative positioning and perspectives of objects in a scene, as well as the viewpoint of the observer. The generated questions were then systematically organized into both a `CSV` and a `JSONL` file, pairing each image with its respective queries. In addition, the dataset could potentially be used to fine-tune VLMs on answering similar types of questions.

Human users' endorsement. We randomly select 20 rendered images from the dataset, along with their captions, and invite volunteer users to rate the captions' correctness with two options (yes or no; Figure 9a). Most responses agree with all captions generated for the rendered images. Grouping captions into three categories: (1) object relations from the viewer's perspective, (2) object relations from the objects' intrinsic perspectives, and (3) object orientations, we find that each category has an average endorsement rate above 93% with low standard



(a) Human user endorsement.



(b) CLIP endorsement.

Figure 9: Average user and CLIP endorsement percentages of captions for each caption category.

errors, indicating strong participant agreement and supporting the toolkit's accuracy and reliability—in fact, the only cases where the participants disagreed were due to a single bookshelf object with a somewhat unclear front view.

CLIP endorsement. The CLIP model (Radford et al., 2021) is known to be fairly capable of recognizing objects; therefore, we also evaluate whether the selected objects can be correctly identified by the model. For each of the 21 preset objects, we render 5 scenes with random orientations and camera configurations, and then we use the CLIP probability over labels as its endorsement level, with the full label set being the 21 object names (Figure 9b). The results show that our present objects are recognized with strong probabilities, serving as evidence that the objects appear in a canonical form.

6 Fine-Tuning Experiments

To further demonstrate the potential of FORG3D, we perform fine-tuning experiments as follows. We synthesize a dataset comprising 31,986 unique rendered images depicting diverse objects and scenes with FORG3D. Each image is paired with contextually relevant questions and answers generated through constructed templates derived from the 3DSR Benchmark, resulting in a dataset of 122,870 question-answer pairs. We then fine-tune the Qwen2-VL-2B-Instruct model (Wang et al., 2024a), which has around 2.2 billion parameters. Due to the computational demands inherent in training models of this size, we utilized Low-Rank Adaptation (LoRA; Hu et al., 2022), significantly reducing computational overhead by limiting up-

State	Dataset Evaluated	Accuracy (overall)	Accuracy (cat. 1)	Accuracy (cat. 2)	Accuracy (cat. 3)
Before Fine-tuning	FORG3D Validation	34.61%	46.12%	52.24%	23.54%
After Fine-tuning	FORG3D Validation	46.33%	42.92%	52.03%	45.34%
After Fine-tuning (enhanced)	FORG3D Validation	49.79%	46.80%	52.03%	50.00%
Before Fine-tuning	3DSR Benchmark	45.37%	58.72%	49.86%	27.41%
After Fine-tuning	3DSR Benchmark	45.66%	53.20%	49.86%	33.82%
After Fine-tuning (enhanced)	3DSR Benchmark	47.00%	52.91%	49.57%	38.48%

Category 1: *front-back categorization* ***Category 2:*** *left-right categorization* ***Category 3:*** *viewpoint-relative reasoning*

Minimal change (<5% difference) | Noticeable decrease (-5% or more)

Noticeable increase (between 5–10%) | Significant increase (10% or more)

Table 1: Performance before and after fine-tuning across datasets and reasoning categories.

dates to a small subset of parameters.

After the fine-tuning procedure, we measure the model's accuracy on questions selected from both the 3DSR dataset and a separate validation set constructed from images generated by the FORG3D that is disjoint from the training set. The questions fall into three categories (Table 1): both front-back and left-right tasks require a binary choice; in contrast, viewpoint-relative reasoning demands locating one object with respect to another (left, right, in front of, or behind).

With fine-tuning, the model's accuracy on the FORG3D validation improves to 46.33% from 34.61%. However, the gains only appear in viewpoint-relative reasoning (23.54% to 45.34%), while the other categories' accuracies decreased slightly. On 3DSR, the overall accuracy exhibits stability, with very minor improvement from 45.37% to 45.66%. However, viewpoint-relative reasoning questions again shows a noteworthy increase, from 27.41% to 33.82%. Conversely, a decline was observed in simpler spatial reasoning questions (front-back categorization), reflecting a potential trade-off as the model adapted to more complex tasks. There was no change in accuracy for the left-right categorization questions (Table 1). The results resonate with those reported by Zhang et al. (2025), where viewpoint-related tasks are identified as challenging problems for VLMs.

Furthermore, we generate a smaller enhanced dataset with 20,652 images and 98,536 question-answer pairs, introducing background variability via the AI background generation pipeline, aiming to improve model robustness by adding noise. By fine-tuning the model with this enhanced dataset, further minor improvements in accuracy are observed: the fine-tuned model reaches an accuracy of 49.79% on the FORG3D validation set, with significant improvements in the viewpoint-relative reasoning category (23.54% to 50.00%) and similar performance on other categories. Furthermore, the fine-tuned model achieves 47% overall accuracy on 3DSR, with viewpoint-relative reasoning accuracy improving by over 11% to 38.48%. However, front-back categorization accuracy again decreases while left-right categorization accuracy remains similar.

Although the overall accuracy improvements observed through fine-tuning are modest, the substantial gains in viewpoint-relative reasoning accuracy are notable and show that the system does have potential. A key limitation, however, is that the training dataset involves only 21 distinct objects, which does not introduce significant variety and may restrict the model's ability to generalize. Future work should focus on refining fine-tuning methods and dataset composition to bolster performance in advanced reasoning tasks without compromising accuracy on simpler spatial categories.

7 Conclusion and Discussion

We have presented FORG3D, a cross-platform 3D rendering toolkit designed to generate high-quality vision-language datasets for spatial reasoning tasks, and demonstrated its potentials through both qualitative demonstrations (§4) and quantitative (§5) evaluations. It offers a user-friendly command-line interface for creating intricate 3D scenes with minimal Blender expertise, allowing researchers in both cognitive science and computer science to focus on the design of their experiments rather than the technical details of rendering. We anticipate FORG3D will facilitate research in both areas.

Acknowledgements. This work is supported in part by a Canada CIFAR AI Chair Award to FS, as well as NSERC RGPIN-2024-04395.

References

Andrea Bender, Sarah Teige-Mocigemba, Annelie Rothe-Wulf, Miriam Seel, and Sieghard Beller. 2020. Being *In Front* Is Good—But Where Is *In Front* ? Preferences for Spatial Referencing Affect Evaluation. *Cognitive Science*, 44(6):e12840.

Blender Team. 2016. *Blender 2.78 Documentation*. Accessed: 2025-03-28.

Blender Team. 2024. *Blender 4.3 Reference Manual*. Accessed: 2025-03-27.

David J. Bryant and Barbara Tversky. 1992. Assessing spatial frameworks with object and direction probes. *Bulletin of the Psychonomic Society*, 30(1):29–32.

David J Bryant, Barbara Tversky, and Nancy Franklin. 1992. Internal and external spatial frameworks for representing described scenes. *Journal of Memory and Language*, 31(1):74–98.

Laura A. Carlson and Shannon R. Van Deman. 2008. Inhibition within a reference frame during the interpretation of spatial language. *Cognition*, 106(1):384–407.

L.A. Carlson-Radvansky and D.E. Irwin. 1994. Reference frame activation during spatial term assignment. *Journal of Memory and Language*, 33(5):646–671.

Boyuan Chen, Zhuo Xu, Sean Kirmani, Brain Ichter, Dorsa Sadigh, Leonidas Guibas, and Fei Xia. 2024a. Spatialvlm: Endowing vision-language models with spatial reasoning capabilities. In *CVPR*.

Boyuan Chen, Zhuo Xu, Sean Kirmani, Brian Ichter, Danny Driess, Pete Florence, Dorsa Sadigh, Leonidas Guibas, and Fei Xia. 2024b. SpatialVLM: Endowing vision-language models with spatial reasoning capabilities. ArXiv:2401.12168 [cs].

An-Chieh Cheng, Hongxu Yin, Yang Fu, Qiushan Guo, Ruihan Yang, Jan Kautz, Xiaolong Wang, and Sifei Liu. 2024. Spatialrgpt: Grounded spatial reasoning in vision language models. In *NeurIPS*.

William G. Hayward and Michael J. Tarr. 1995. Spatial language and spatial representation. *Cognition*, 55(1):39–84.

Yining Hong, Haoyu Zhen, Peihao Chen, Shuhong Zheng, Yilun Du, Zhenfang Chen, and Chuang Gan. 2023. 3d-llm: Injecting the 3d world into large language models. In *NeurIPS*.

Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. 2022. Lora: Low-rank adaptation of large language models. In *ICLR*.

Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. 2017. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *CVPR*.

Amita Kamath, Jack Hessel, and Kai-Wei Chang. 2023. What's "up" with vision-language models? investigating their struggle with spatial reasoning. In *EMNLP*.

Barbara Landau and Ray Jackendoff. 1993. Whence and whither in spatial language and spatial cognition? *Behavioral and Brain Sciences*, 16(2):255–265.

Stephen C. Levinson. 2003. *Space in language and cognition: Explorations in cognitive diversity*, 1 edition. Cambridge University Press.

Manling Li, Shiyu Zhao, Qineng Wang, Kangrui Wang, Yu Zhou, Sanjana Srivastava, Cem Gokmen, Tony Lee, Erran Li Li, Ruohan Zhang, et al. 2024. Embodied agent interface: Benchmarking llms for embodied decision making. In *NeurIPS*.

Peggy Li and Lila Gleitman. 2002. Turning the tables: Language and spatial reasoning. *Cognition*, 83(3):265–294.

Fangyu Liu, Guy Emerson, and Nigel Collier. 2023a. Visual spatial reasoning. *Transactions of the Association for Computational Linguistics (TACL)*, 11:635–651.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023b. Visual instruction tuning. In *NeurIPS*.

Runtao Liu, Chenxi Liu, Yutong Bai, and Alan L Yuille. 2019. Clevr-ref+: Diagnosing visual reasoning with referring expressions. In *CVPR*.

G.D. Logan. 1995. Linguistic and Conceptual Control of Visual Spatial Attention. *Cognitive Psychology*, 28(2):103–174.

Wufei Ma, Haoyu Chen, Guofeng Zhang, Celso M de Melo, Alan Yuille, and Jieneng Chen. 2024. 3dsr-bench: A comprehensive 3d spatial reasoning benchmark. *arXiv preprint arXiv:2412.07825*.

Jiayuan Mao, Xuelin Yang, Xikun Zhang, Noah Goodman, and Jiajun Wu. 2022. CLEVRER-Humans: Describing physical and causal events the human way. In *NeurIPS*.

Michael Ogezi and Freda Shi. 2025. SpaRE: Enhancing spatial reasoning in vision-language models with synthetic data. In *ACL*.

Jonas Pfeiffer, Gregor Geigle, Aishwarya Kamath, Jan-Martin O Steitz, Stefan Roth, Ivan Vulić, and Iryna Gurevych. 2022. xGQA: Cross-lingual visual question answering. In *Findings of ACL*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *ICML*.

Terry Regier and Laura A Carlson. 2001. Grounding spatial language in perception: an empirical and computational investigation. *Journal of experimental psychology: General*, 130(2):273.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. In *CVPR*.

Fatemeh Shiri, Xiao-Yu Guo, Mona Far, Xin Yu, Reza Haf, and Yuan-Fang Li. 2024. An empirical analysis on spatial reasoning capabilities of large multimodal models. In *EMNLP*.

Pablo Villalobos, Anson Ho, Jaime Sevilla, Tamay Besiroglu, Lennart Heim, and Marius Hobbhahn. 2024. Position: Will we run out of data? limits of llm scaling based on human-generated data. In *ICML*.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. 2024a. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*.

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

Xingrui Wang, Wufei Ma, Tiezheng Zhang, Celso M de Melo, Jieneng Chen, and Alan Yuille. 2025. Pulsecheck457: A diagnostic benchmark for comprehensive spatial reasoning of large multimodal models. *arXiv preprint arXiv:2502.08636*.

Kexin Yi, Chuang Gan, Yunzhu Li, Pushmeet Kohli, Jiajun Wu, Antonio Torralba, and Joshua B Tenenbaum. 2020. CLEVRER: Collision events for video representation and reasoning. In *ICLR*.

Zheyuan Zhang, Fengyuan Hu, Jayjun Lee, Freda Shi, Parisa Kordjamshidi, Joyce Chai, and Ziqiao Ma. 2025. Do vision-language models represent space and how? Evaluating spatial frame of reference under ambiguities. In *ICLR*.

A Limitations

Despite its versatility, the current implementation of FORG3D has several limitations:

1. User interface and usability: Currently, the toolkit is primarily operated via command-line inputs, which may deter users unfamiliar with scripting. Developing an intuitive graphical user interface could enhance accessibility.
2. Support for multiple objects in one scene: The toolkit is designed to render scenes containing only two objects, focusing on their relative spatial configurations. Expanding the tool to support scenes with more than two objects would better reflect real-world environments.

B Future Improvements

Our roadmap for extending the FORG3D toolkit in the future involves implementing a detailed strategy to effectively manage the complexity of rendering multi-object scenes. Specifically, we plan to implement advanced:

(a) Positioning logic: develop a hierarchical positioning system that extends current pairwise logic to efficiently handle positioning for n-body collections. This will involve spatial partitioning techniques to systematically manage relationships among multiple objects.

(b) Occlusion prevention: introduce real-time occlusion checking for multiple objects using depth analysis to ensure they remain visible.

(c) Combinatorial management: use AI to automatically discard redundant or visually similar scenes, reducing the vast number of possible arrangements for multiple objects and improving overall efficiency.

By addressing the limitations and implementing the proposed enhancements, future iterations of the toolkit can further enhance its role as a robust platform for synthetic spatial reasoning dataset generation, advancing scientific research in both cognitive science and machine learning.