

Making 3D Scenes Interactive

Mu-sheng Lin

vrlin@stanford.edu

Addison Jadwin

ajadwin@stanford.edu

Tatiana Veremeenko

tataver@stanford.edu

Abstract

Interactive 3D environments are key for training and evaluation of embodied AI agents, which makes availability of realistic large-scale 3D datasets critical for future advances in the area. However, modern state-of-the-art simulation databases rarely contain more than a hundred interactive scenes [26][19][14], because manual scene creation is time-consuming and expensive. While there have been efforts to generate simulation scenes procedurally from scratch [5], the outcomes of training AI agents in these environments are unsatisfactory [12], likely because these solutions use overly simplistic rules, leading to unrealistic generations. Meanwhile, 3D reconstruction datasets built by scanning real environments may contain thousands of realistic scenes [4][42][40], while lacking physical object annotations necessary for interactivity.

In this project we introduce a solution that leverages abundant 3D reconstruction data available online for procedural scene generation in two stages. First, we use Grounded SAM [33], a powerful open-set object detection and semantic segmentation tool to segment and annotate the input 3D mesh with user-defined object categories. Second, we find the nearest database object (similar both semantically and in terms of orientation) for every detected object using CLIP [28] image representations and class labels predicted at the previous step. This process may be used to convert any 3D mesh scene (either reconstructed or generated) into an interactive environment by repopulating it with richly annotated database objects replacing their original mesh counterparts.

1. Introduction

Modern robotics research and engineering benefit from scale and availability of quality virtual simulation data, such as indoor/outdoor interactive scenes. Embodied AI agents can be trained and evaluated in virtual environments for a range of tasks because it’s a safe, cheap and scalable alternative to real-world training. Virtual training is also fast, parallelizable and allows more control and reproducibility. However, creating simulation scenes is labor-intensive

and costly as each scene has to be modeled and annotated by humans, especially when it comes to interactive scenes modeling everyday household tasks. If we want an embodied AI agent trained in a simulation to successfully operate in the real world, realism of the scene layout and physical properties of the objects it interacts with become critical. BEHAVIOR-1K [19] is a recently published embodied AI benchmark that addresses these needs with a database of 1,000 pre-sampled activities, 50 scenes, and more than 9,000 objects. While enjoying realistic scene layouts and advanced physical simulation based on the OmniGibson technology, it’s still far from large scale.

In this project we take on the task of converting a 3D indoor scene representation into an interactive OmniGibson scene filled with richly annotated objects from the BEHAVIOR-1K database. Given the availability of large 3D scene reconstruction datasets [4][42][40] and recent advances in visual-language models that make automatic object detection and semantic segmentation possible [13][28][21], this approach promises a reliable and universal solution for the training data scarcity challenge. Our suggested pipeline consists of two stages. First, we take a 3D scene mesh representation together with multiview 2D scans of the scene and perform semantic segmentation with Grounding SAM [33] approach, obtaining a segmented 3D mesh as the result. Second, we isolate objects from the segmented scene and replace them with nearest neighbors from the BEHAVIOR-1K object database, using distance between CLIP embeddings of 2D renders of the objects taken from the same angle. Each of these stages comes with its unique set of challenges we need to address.

2. Related Work

3D representations are ways of modeling and storing three-dimensional objects/scenes in a form that computers can manipulate and render. The most common 3D representations are 1) voxel [15], 2) polygon mesh [37] and 3) point cloud [17]. Voxels in 3D space are the similar concept as pixels in 2D space while polygon meshes are vertices with color and face sets of polygon (typically with triangles). The state-of-the-art method apply 3D Gaussian splatting [11] with point cloud representation, which renders real

time and high resolution 3D scenes.

Visual-language models or multimodels [24] move one step forward from pure Large Language Model (LLM) [29] [2] by taking images as the input. The earlier work of representations by unsupervised Contrastive Predictive Coding in [38] inspires the Contrastive Language-Image Pre-training (CLIP) [28], which is trained by large scale of datasets consisting of pair of text and images from internet. Grounding DINO [21] extends the concept of cross attention between image tokens and text tokens, which achieves the open set object detection capability. Similar cross attention mechanism but with point and box prompt embeddings in SAM [13] achieves the major breakthrough of image segmentation. In combination of the two methods, Grounding SAM [33] was proposed to leverage the Grounding DINO capabilities to generate the box prompt for SAM to produce more accurate mask annotations.

3D semantic segmentation of point cloud defines the process of partitioning the 3D representation (e.g. point cloud) into multiple sets of points, each identifying a different object or part of the scene at the point-level. With benefit from the recent advance in segmentation (SAM [13]), the authors in [43] proposes to iteratively merge the 2D segmentation masks generated from multiviews of the point cloud from a given scene. In [41], the idea promotes projecting located point from 3D input scenes to 2D frames as the pixel prompts to the SAM [13], which has better frame consistency and surpasses human-level annotations. Improvement of 3D consistency is achieved in [44] by evaluating the primitive affinity scores among the 2D posed masks from SAM and merging progressively. Alternately, authors in [27] proposed to learn hierarchical semantics using SAM.

3D object bounding boxes could be either the extension task from 2D object detection with bounding boxes or the consequential results built on 3D segmentation tasks. 2D object detection with bounding boxes could be achieved by either traditional convolution neural network (CNN) based methods ([9] [8] [32] [30] [31]) or recent transformer base methods ([3] [7] [45]). The concept has been extended to 3D with multiple viewpoints (e.g. [23]). For 3D objects in motion, Multi-Frame Clustering (MFC) has been proposed in [39].

3D Awareness Merging Algorithms are designed to merge 2D images into 3D mesh/point clouds with geometry consistency. Structure from Motion (SfM) considers 3D reconstruction from unordered image collections [25]. In [35], COLMAP is proposed, which consists of 1) feature extraction (e.g. SIFT [22] for keypoints), 2) matching and 3) geometric verification in the Correspondence Search among the images. More recent approaches uses SAM [33] to identify the same object masks among images (refer to SAM3D [43]) and then applies the bidirectional merging to generate the 3D masks. SAMPro3D [41] suggested to lever-

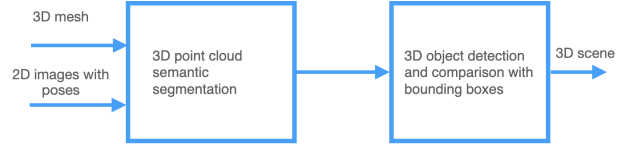


Figure 1: Functional diagram of the two stage pipeline.

age the SAM’s point prompt capability in 2D images with 3D initial points projecting to 2D as point prompt inputs for SAM, which has better geometry consistency. SAI3D [44] groups 3D points in the point cloud as 3D primitives by normal-based graph cut and then projects to 2D images, segments with SAM, and merges back to 3D with affinity scores in 2D (cosine similarity) in single view (weighted sum for multi-view cases). Authors in [1] analyzed the 3D awareness by 1) encode the 3D structure and 2) consistency represent the surface across view via the features extracted from the visual foundation models and show the limitations of current visual models.

3. Approach

As mentioned in Sec. 1, our pipeline consists of two main stages (Fig. 1):

- 3D semantic segmentation
- 3D database object identification and replacement

We provide the details of both stages as well as intermediate steps below.

3.1. 3D semantic segmentation

Our solution for the first stage is based on Grounding DINO and SAM models that provide powerful capabilities for open-set object detection and prompt-based segmentation, followed by application of a simple yet effective SAM3D framework.

We start with object categories from the BEHAVIOR-1K dataset as text prompts and apply Grounding DINO algorithm to the multiview 2D scans to detect objects in the given scene that may have suitable counterparts from the same categories in BEHAVIOR-1K. For simplicity, we manually filter out object categories that are not present in the given scene, a task that may be performed using visual-language models in the future. Grounding DINO outputs a set of labeled bounding boxes that serves as prompts for the SAM model that produces a binary segmentation mask per each class per image. We then combine the per class masks, producing a single semantic segmentation mask per image. As the last step we utilize

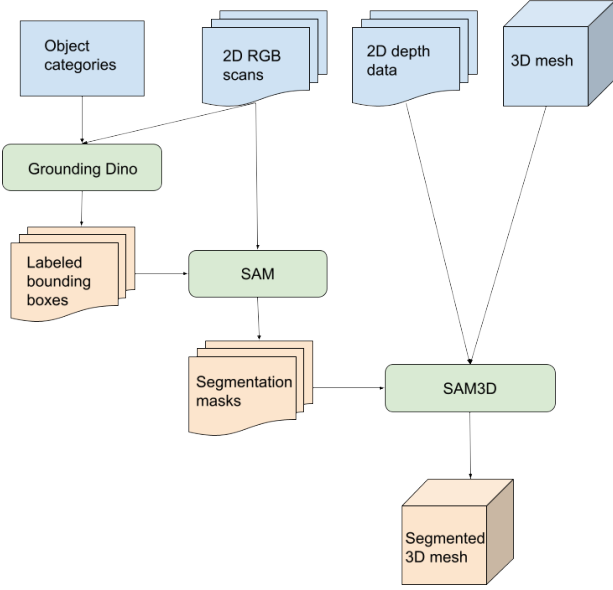


Figure 2: Functional diagram of the first stage pipeline.

the SAM3D tool (codes available at <https://github.com/Pointcept/SegmentAnything3D>) that uses original 2D depth data to project the 2D segmentation masks back to the original 3D scene.

3.2. Object Identification and Replacement

The next stage of the pipeline after semantic segmentation involves matching the isolated object(s) from the 3d mesh to an object instance and orientation pair in the BEHAVIOR-1K dataset, and finally replacing the original object with the interactive one. To do this, we choose a camera position and angle in the 3d mesh to get a 'snapshot' of the objects in the scene that we can replace. Then, we first generate two altered meshes using the original semantically segmented one: (1) a mesh where everything is filtered out except for the object we want to replace, and (2) a mesh where the object itself is filtered out and everything else remains. These meshes can be visualized in Fig 10 and Fig. 11 with first stage's 3D segmentation masks.

We then use mesh (1) to get a 2d image of the object and compare it to 2d images of the BEHAVIOR-1K objects that have the same label. The BEHAVIOR-1K objects are rendered at a variety of orientations but all have the same camera pose as the original scene. To identify the correct object and orientation, we first match the 3d mesh object to one of the object instances in the BEHAVIOR-1K dataset. For instance, if we know the object's label is 'straight_chair', we render one image of each instance of

straight chairs in the BEHAVIOR-1K dataset and match the 3d mesh object to one of the instances. Then, as a second step, we match the 3d mesh object to one of multiple orientations of the BEHAVIOR-1K object instance, as to account for the fact that objects in the 3d mesh may not always be facing the same direction in relation to the camera pose.

To do the matching process described above, we leverage large vision/language models CLIP [28] and GPT-4o. With CLIP, we get an embedding of each of the objects (both the 3d mesh object and the BEHAVIOR-1K objects) and compute cosine similarity between the 3d mesh object and the BEHAVIOR-1K objects to match them. When matching the 3d mesh object to the correct orientation, we also use PCA to reduce the dimensionality of the embeddings of each image. In doing this we aim to distill the embeddings such that the remaining dimensions reflect orientation/rotation information, since this is the only thing varying between the dataset images at this stage in the pipeline. The original object's embedding is then projected onto these dimensions to make the comparison. With GPT-4o, rather than comparing image embeddings directly, we prompt GPT to compare the orientation of the 3d mesh image to the orientations of each of the BEHAVIOR-1k images and choose the BEHAVIOR-1k image that matches it best.

The prompt we use for GPT:

```

You are an object orientation matcher.
You are given one image of a novel
object (first image) and a few
images of a known object in different
orientations (subsequent images).
Your job is to tell me which of the
images of the known objects is closest
in orientation to the novel object
image. Please complete the task with
the attached images. You should not
write or use any code, but should
instead simply look at the images.
Also, you should pay attention to which
way (e.g. front, right, front left,
etc) the chair is facing to inform your
decision.
  
```

Finally, we use mesh (2) to build a new scene where the original object is replaced by the BEHAVIOR-1k object. We use the best instance and angle computed (in relation to the camera pose) to place the object into the scene. We also save the adjusted bounding box (center and extents) as well as orientation to later create a fully interactive OmniGibson environment.

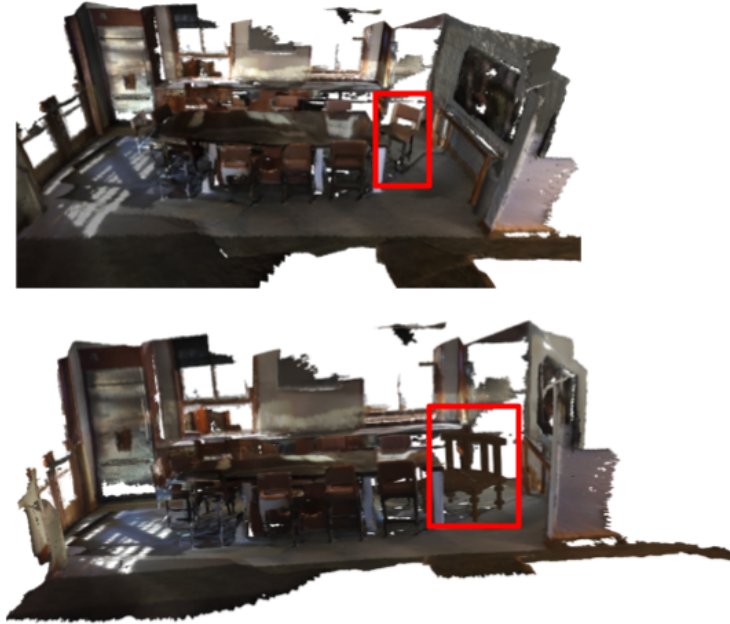


Figure 3: The original scene vs the scene with a BEHAVIOR-1K object replacing the original one. The object in question is highlighted in red.

4. Dataset and Features

As described in Sec. 1, the focus of this project is on the indoor scenes. Two datasets: 1) ScanNet [4] and 2) BEHAVIOR-1K datasets [19] are applied to our experiments. The former is used for indoor scenes segmentation (one typical scene demonstrated in Fig. 4) while the latter is to generate interactive 3D object by replacing the original ScanNet 3D object with objects from BEHAVIOR-1K datasets (one Omnigibson supported object with states shown in Fig. 5).

We select ScanNet datasets for our experiments with instance-level indoor RGB-D data (depth is 16-bit unsigned short values), which consists of 2.5M views in 1513 scenes. The datasets contains both 3D meshes with corresponding 2D multi-view images and semantic annotations. In addition, the ScanNet datasets have been used in many 3D segmentation researches [43] [41] [44].

To achieve the goals of interactive 3D indoor environments, as introduced in 1, BEHAVIOR-1K datasets is adopted since it contains more than 9,000 annotated objects with physical (e.g., friction, mass, articulation) and semantic (e.g., category) properties. Particularly, Omnigibson provides, beyond [18], necessary functionalities to support more realistic interactive 1000 activities (continuous



Figure 4: One typical indoor scene from ScanNet [4].

extended states such as thermal effect, deformables, transparency and so forth).

Here is detail of applying the ScaNet and BEHAVIOR-1K datasets to our experiments:

- For the first stage of the pipeline, we start with ScanNet dataset [4], that provides 3D reconstructions of indoor scenes, annotated with camera poses and semantic seg-

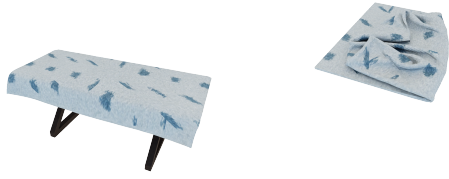


Figure 5: One Omnigibson supported object with states from BEHAVIOR-1K [19]: (left) draped and (right) folded.

mentation.

- For the second stage of the pipeline, the images of the filtered 3D object/mesh (by SAM3D masks) will be compared to the images of objects in the OmniGibson BEHAVIOR-1K datasets.

5. Experiment Results and Discussion

5.1. 3D Semantic Segmentation Results

We ran our pipeline described in Sec 3.1 for individual ScanNet scenes in two modes: binary segmentation, using a single class label as text prompt for Grounding DINO and multi-class segmentation, running the algorithm for several classes and combining the resulting masks together. We assessed the performance of the chosen approach qualitatively by rendering the resulting labeled 3D meshes and comparing them with the ground truth annotations (Fig. 6). An example of single class segmentation is shown at Fig. 8, and Fig 7 demonstrates a result of running multi-class segmentation using a list of objects presented in the original scene as text prompts. Looking at these renders, we can make a couple of important observations:

- Single-class segmentation results are visibly better, while not perfect. Multi-class segmentation results in a spotty mask that can only approximately differentiate very different areas (such as chairs vs. refrigerator), but is very inconsistent in more similar parts of the scene (refrigerator vs. cabinets)
- Single-class segmentation works better, but the resulting mask is still spotty, sometimes covering the unexpected areas, for example, the lower part of the refrigerator in a suspiciously straight line. These artifacts may be caused by inconsistencies in depth data that is used by SAM3D algorithm to lift the 2D segmentation masks back to 3D mesh (See Fig. 9 for an example)

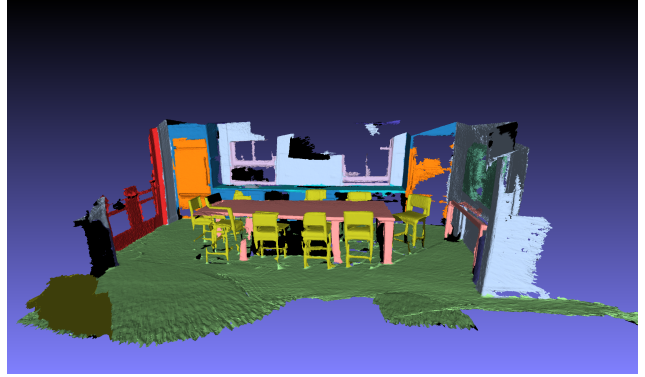


Figure 6: Ground truth segmentation mask.

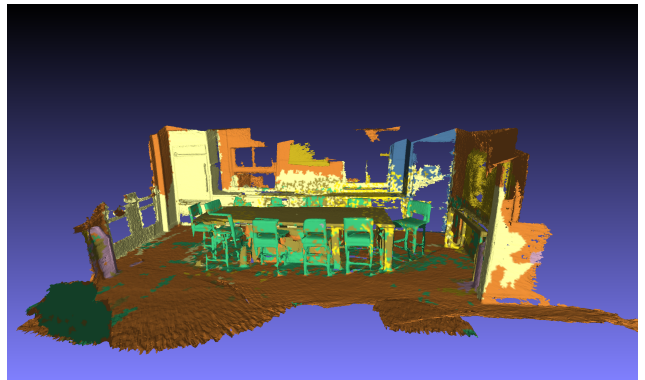


Figure 7: Multi-class segmentation result. Text prompts: ['refrigerator', 'cabinet', 'doors', 'chair', 'sink', 'kitchen cabinets', 'tv', 'armchair', 'trash can', 'table']

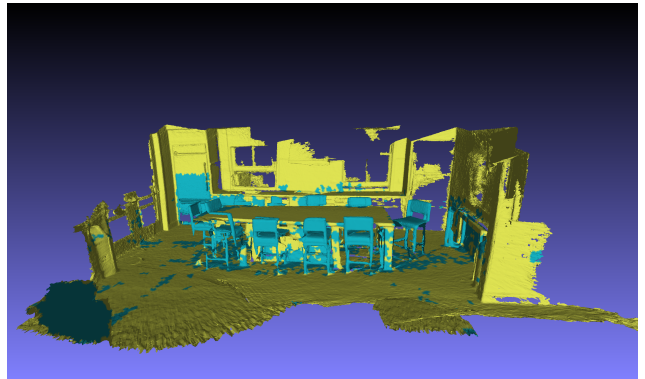
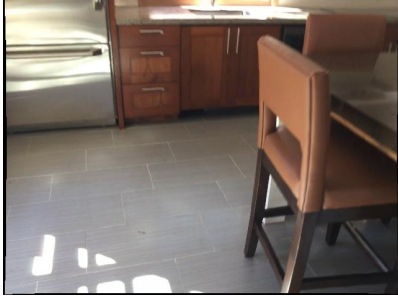


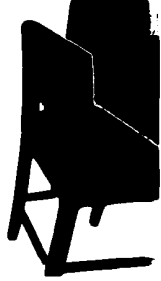
Figure 8: Binary segmentation result. Text prompt: 'chair'

5.2. Object Identification And Replacement Results

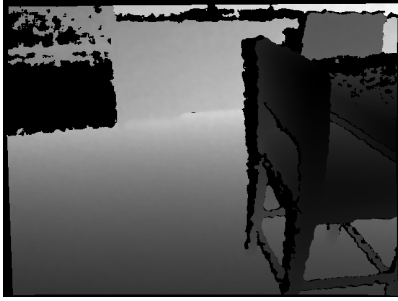
Fig. 3 shows an example end-to-end result of replacing a 3d mesh object with a BEHAVIOR-1k object. Generally, CLIP does very well in matching the 3d mesh object



(a) RGB frame



(b) SAM segmentation mask



(c) Depth scan

Figure 9: Depth artifacts. Figure (a) is the original RGB frame, showing the placement of a chair and a fridge. SAM segmentation mask on figure (b) demonstrates that SAM was able to properly distinguish chairs from the refrigerator. However, on the depth scan (c) similar depths values are erroneously assigned to the chairs and parts of the refrigerator.

to a similar BEHAVIOR-1k instance in its respective category (see example in Fig. 15). It also does reasonably well in matching the object to its respective orientation (see example in Fig. 16) when we apply PCA to the embeddings. Table 1 shows the accuracy of CLIP alone, CLIP with PCA, and GPT-4o in matching orientations on a small subset (5 instances) of straight chairs from the BEHAVIOR-1k dataset. As shown, CLIP seems to rely on PCA to focus the embeddings on orientation information, and GPT-4o doesn't perform very well. It should be noted, however, that most of CLIP's errors were off by only one rotation (45

degrees in this case), indicating this deficiency could potentially be ameliorated by higher quality meshes/images or other variations of dimensionality reduction, etc. See the discussion section for more details on failure cases.

Model	Accuracy
CLIP	40%
CLIP with PCA	80%
GPT-4o	40%

Table 1: Comparison of models for matching 3d mesh objects to orientations of the corresponding BEHAVIOR-1k object instance. This is tested on a small set of 5 different instances of the straight chair BEHAVIOR-1k object category.

5.3. Discussion and limitations

As shown in Fig. 11 (left), the mesh of the chair is fragmented (missing some legs) after we apply the masks generated by SAM while some parts of the legs' mesh is left in the filtered scene as shown in the right of Fig. 11. This is not the issue from SAM but the limitation from the original datasets. As shown in Fig. 12, in the multiview images, the legs of the chair are occluded by the table, so SAM could not segment in the 2D images and hence the 3D mask is missing these parts. In SAI3D [44], this issue has been addressed by the affinity scores to some extent.

One other limitation we faced was CLIP/GPT's performance in matching the 3d mesh object to orientations of the BEHAVIOR-1k objects. While overall we managed to get satisfactory performance with CLIP and PCA, it is not consistent across all objects/instances. With CLIP, most of the errors are off by one rotation of 45 degrees, so better performance may certainly be attainable, perhaps with a more granular set of orientations to both make errors less drastic and to increase the dimensionality of the PCA-reduced embeddings since there would be more embeddings in the dataset. GPT's errors seem mainly related to the model confusing which image is which or whether it should make orientation comparisons from the chair or the viewer's perspective. Better prompt engineering may thus yield more consistent results if CLIP still cannot achieve desirable performance.

6. Conclusion and Future Work

In this project we introduced a general framework for creating new realistic simulation scenes based on existing 3D scene representations, leveraging the recent developments in visual-language representation modeling and robotic interactive benchmarks. We explored applications of Grounding SAM to the 3D semantic segmentation tasks and CLIP embedding capabilities for most similar object



Figure 10: Visualized 3D masks from SAMPro3D: (left) chair 3D mask and (right) the whole scene without chair 3D mask. Notice that the masks of non-target things are zeroed out by setting the RGB as white color with zero alpha.



Figure 11: Images after SAM masks: (left) the whole scene, except for the chair, is filtered out. (right) the chair is filtered out.

identification. While both parts of our pipeline may require further work to achieve end-to-end performance, we can see the promise of our approach for future applications. One important takeaway from this work we’d like to emphasize is the critical importance of the original 3D mesh data that may remove the dependency on incomplete and potentially flawed 2D scans.

Another potential direction of future work is text-to-3D generation. In recent research, efforts are made to extend the text-to-2D images [34] to text-to-3D scenes [36] [20] [16]. Our pipeline could be extended further to generate the whole 3D scenes from text prompts, which enables vastly diversified robotic simulation in the wild. Particularly, 3D Gaussian splatting (GS) [11] due to its real-time performance and high quality images (refer to Appendix A for some pilot studies). Combining 3D segmentation with language grounding with 3D GS [27] could also be interesting future study.

References

- [1] M. E. Banani, A. Raj, K.-K. Maninis, A. Kar, Y. Li, M. Rubinstein, D. Sun, L. Guibas, J. Johnson, and V. Jampani. Probing the 3d awareness of visual foundation models, 2024.
- [2] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners, 2020.
- [3] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko. End-to-end object detection with transformers. In A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, editors, *Computer Vision – ECCV 2020*, pages 213–229, Cham, 2020. Springer International Publishing.
- [4] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner. Scannet: Richly-annotated 3d reconstruc-

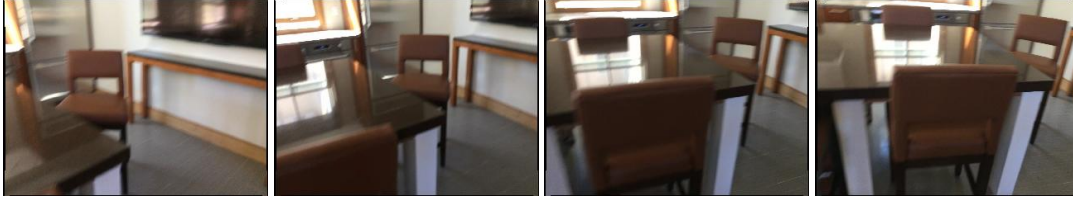


Figure 12: The legs of the chair are occluded in the original video photos.

- tions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2017.
- [5] M. Deitke, E. VanderBilt, A. Herrasti, L. Weihs, J. Salvador, K. Ehsani, W. Han, E. Kolve, A. Farhadi, A. Kembhavi, and R. Mottaghi. Proctor: Large-scale embodied ai using procedural generation, 2022.
- [6] P. Engstler, A. Vedaldi, I. Laina, and C. Rupprecht. Invisible stitch: Generating smooth 3d scenes with depth inpainting. In *Arxiv*, 2024.
- [7] Y. Fang, B. Liao, X. Wang, J. Fang, J. Qi, R. Wu, J. Niu, and W. Liu. You only look at one sequence: Rethinking transformer in vision through object detection. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 26183–26197. Curran Associates, Inc., 2021.
- [8] R. Girshick. Fast r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.
- [9] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 580–587, 2014.
- [10] L. Höllein, A. Cao, A. Owens, J. Johnson, and M. Nießner. Text2room: Extracting textured 3d meshes from 2d text-to-image models, 2023.
- [11] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), July 2023.
- [12] M. Khanna, Y. Mao, H. Jiang, S. Hareesh, B. Shacklett, D. Batra, A. Clegg, E. Undersander, A. X. Chang, and M. Savva. Habitat synthetic scenes dataset (hssd-200): An analysis of 3d scene scale and realism tradeoffs for objectgoal navigation, 2023.
- [13] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick. Segment anything, 2023.
- [14] E. Kolve, R. Mottaghi, W. Han, E. VanderBilt, L. Weihs, A. Herrasti, M. Deitke, K. Ehsani, D. Gordon, Y. Zhu, A. Kembhavi, A. Gupta, and A. Farhadi. Ai2-thor: An interactive 3d environment for visual ai, 2022.
- [15] K. Kutulakos and S. Seitz. A theory of shape by space carving. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 1, pages 307–314 vol.1, 1999.
- [16] H.-H. Lee, M. Savva, and A. X. Chang. Text-to-3d shape generation, 2024.
- [17] M. Levoy and T. Whitted. The use of points as a display primitive. 2000.
- [18] C. Li, F. Xia, R. Martín-Martín, M. Lingelbach, S. Srivastava, B. Shen, K. E. Vainio, C. Gokmen, G. Dharan, T. Jain, A. Kurenkov, K. Liu, H. Gweon, J. Wu, L. Fei-Fei, and S. Savarese. igibson 2.0: Object-centric simulation for robot learning of everyday household tasks. In A. Faust, D. Hsu, and G. Neumann, editors, *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pages 455–465. PMLR, 08–11 Nov 2022.
- [19] C. Li, R. Zhang, J. Wong, C. Gokmen, S. Srivastava, R. Martín-Martín, C. Wang, G. Levine, W. Ai, B. Martinez, H. Yin, M. Lingelbach, M. Hwang, A. Hiranaka, S. Garland, A. Aydin, S. Lee, J. Sun, M. Anvari, M. Sharma, D. Bansal, S. Hunter, K.-Y. Kim, A. Lou, C. R. Matthews, I. Villa-Renteria, J. H. Tang, C. Tang, F. Xia, Y. Li, S. Savarese, H. Gweon, C. K. Liu, J. Wu, and L. Fei-Fei. Behavior-1k: A human-centered, embodied ai benchmark with 1,000 everyday activities and realistic simulation, 2024.
- [20] C.-H. Lin, J. Gao, L. Tang, T. Takikawa, X. Zeng, X. Huang, K. Kreis, S. Fidler, M.-Y. Liu, and T.-Y. Lin. Magic3d: High-resolution text-to-3d content creation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.
- [21] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, C. Li, J. Yang, H. Su, J. Zhu, and L. Zhang. Grounding dino: Marrying dino with grounded pre-training for open-set object detection, 2023.
- [22] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, 2004.
- [23] H. Lu, J. Tang, X. Xu, X. Cao, Y. Zhang, G. Wang, D. Du, H. Chen, and Y. Chen. Scaling multi-camera 3d object detection through weak-to-strong eliciting, 2024.
- [24] OpenAI, J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altschmidt, S. Altman, S. Anadkat, R. Avila, I. Babuschkin, S. Balaji, V. Balcom, P. Baltescu, H. Bao, M. Bavarian, J. Belgum, I. Bello, J. Berdine, G. Bernadett-Shapiro, C. Berner, L. Bogdonoff, O. Boiko, M. Boyd, A.-L. Brakman, G. Brockman, T. Brooks, M. Brundage, K. Button, T. Cai, R. Campbell, A. Cann, B. Carey, C. Carlson, R. Carmichael, B. Chan, C. Chang, F. Chantzis, D. Chen, S. Chen, R. Chen, J. Chen, M. Chen, B. Chess, C. Cho, C. Chu, H. W.

- Chung, D. Cummings, J. Currier, Y. Dai, C. Decareaux, T. Degry, N. Deutsch, D. Deville, A. Dhar, D. Dohan, S. Dowling, S. Dunning, A. Ecoffet, A. Eleti, T. Eloundou, D. Farhi, L. Fedus, N. Felix, S. P. Fishman, J. Forte, I. Fulford, L. Gao, E. Georges, C. Gibson, V. Goel, T. Gogineni, G. Goh, R. Gontijo-Lopes, J. Gordon, M. Grafstein, S. Gray, R. Greene, J. Gross, S. S. Gu, Y. Guo, C. Hallacy, J. Han, J. Harris, Y. He, M. Heaton, J. Heidecke, C. Hesse, A. Hickey, W. Hickey, P. Hoeschele, B. Houghton, K. Hsu, S. Hu, X. Hu, J. Huizinga, S. Jain, S. Jain, J. Jang, A. Jiang, R. Jiang, H. Jin, D. Jin, S. Jomoto, B. Jonn, H. Jun, T. Kaftan, Łukasz Kaiser, A. Kamali, I. Kanitscheider, N. S. Keskar, T. Khan, L. Kilpatrick, J. W. Kim, C. Kim, Y. Kim, J. H. Kirchner, J. Kiros, M. Knight, D. Kokotajlo, Łukasz Kondruciuk, A. Kondrich, A. Konstantinidis, K. Kosic, G. Krueger, V. Kuo, M. Lampe, I. Lan, T. Lee, J. Leike, J. Leung, D. Levy, C. M. Li, R. Lim, M. Lin, S. Lin, M. Litwin, T. Lopez, R. Lowe, P. Lue, A. Makanju, K. Malfacini, S. Manning, T. Markov, Y. Markovski, B. Martin, K. Mayer, A. Mayne, B. McGrew, S. M. McKinney, C. McLeavey, P. McMillan, J. McNeil, D. Medina, A. Mehta, J. Menick, L. Metz, A. Mishchenko, P. Mishkin, V. Monaco, E. Morikawa, D. Mossing, T. Mu, M. Murati, O. Murk, D. Mély, A. Nair, R. Nakano, R. Nayak, A. Neelakantan, R. Ngo, H. Noh, L. Ouyang, C. O’Keefe, J. Pachocki, A. Paino, J. Palermo, A. Pantuliano, G. Parascandolo, J. Parish, E. Parparita, A. Passos, M. Pavlov, A. Peng, A. Perelman, F. de Avila Belbute Peres, M. Petrov, H. P. de Oliveira Pinto, Michael, Pokorny, M. Pokrass, V. H. Pong, T. Powell, A. Power, B. Power, E. Proehl, R. Puri, A. Radford, J. Rae, A. Ramesh, C. Raymond, F. Real, K. Rimbach, C. Ross, B. Rotsted, H. Roussez, N. Ryder, M. Saltarelli, T. Sanders, S. Santurkar, G. Sastry, H. Schmidt, D. Schnurr, J. Schulman, D. Selsam, K. Sheppard, T. Sherbakov, J. Shieh, S. Shoker, P. Shyam, S. Sidor, E. Sigler, M. Simens, J. Sitkin, K. Slama, I. Sohl, B. Sokolowsky, Y. Song, N. Staudacher, F. P. Such, N. Summers, I. Sutskever, J. Tang, N. Tezak, M. B. Thompson, P. Tillet, A. Tootoonchian, E. Tseng, P. Tuggle, N. Turley, J. Tworek, J. F. C. Uribe, A. Vallone, A. Vijayvergiya, C. Voss, C. Wainwright, J. J. Wang, A. Wang, B. Wang, J. Ward, J. Wei, C. Weinmann, A. Welihinda, P. Welinder, J. Weng, L. Weng, M. Wiethoff, D. Willner, C. Winter, S. Wolrich, H. Wong, L. Workman, S. Wu, J. Wu, M. Wu, K. Xiao, T. Xu, S. Yoo, K. Yu, Q. Yuan, W. Zaremba, R. Zellers, C. Zhang, M. Zhang, S. Zhao, T. Zheng, J. Zhuang, W. Zhuk, and B. Zoph. Gpt-4 technical report, 2024.
- [25] O. Ozyesil, V. Voroninski, R. Basri, and A. Singer. A survey of structure from motion, 2017.
- [26] X. Puig, E. Undersander, A. Szot, M. D. Cote, T.-Y. Yang, R. Partsey, R. Desai, A. W. Clegg, M. Hlavac, S. Y. Min, V. Vondruš, T. Gervet, V.-P. Berges, J. M. Turner, O. Maksymets, Z. Kira, M. Kalakrishnan, J. Malik, D. S. Chaplot, U. Jain, D. Batra, A. Rai, and R. Mottaghi. Habitat 3.0: A co-habitat for humans, avatars and robots, 2023.
- [27] M. Qin, W. Li, J. Zhou, H. Wang, and H. Pfister. Langsplat: 3d language gaussian splatting. *arXiv preprint arXiv:2312.16084*, 2023.
- [28] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. Learning transferable visual models from natural language supervision, 2021.
- [29] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. Language models are unsupervised multitask learners. 2019.
- [30] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016.
- [31] D. Reis, J. Kupec, J. Hong, and A. Daoudi. Real-time flying object detection with yolov8, 2023.
- [32] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.
- [33] T. Ren, S. Liu, A. Zeng, J. Lin, K. Li, H. Cao, J. Chen, X. Huang, Y. Chen, F. Yan, Z. Zeng, H. Zhang, F. Li, J. Yang, H. Li, Q. Jiang, and L. Zhang. Grounded sam: Assembling open-world models for diverse visual tasks, 2024.
- [34] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models, 2022.
- [35] J. L. Schönberger and J.-M. Frahm. Structure-from-motion revisited. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4104–4113, 2016.
- [36] J. Tang. Stable-dreamfusion: Text-to-3d with stable-diffusion, 2022. <https://github.com/ashawkey/stable-dreamfusion>.
- [37] G. Turk and M. Levoy. Zippered polygon meshes from range images. In *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH ’94*, page 311–318, New York, NY, USA, 1994. Association for Computing Machinery.
- [38] A. van den Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding, 2019.
- [39] H. Wu, S. Zhao, X. Huang, C. Wen, X. Li, and C. Wang. Commonsense prototype for outdoor unsupervised 3d object detection, 2024.
- [40] F. Xia, A. R. Zamir, Z.-Y. He, A. Sax, J. Malik, and S. Savarese. Gibson env: real-world perception for embodied agents. In *Computer Vision and Pattern Recognition (CVPR), 2018 IEEE Conference on*. IEEE, 2018.
- [41] M. Xu, X. Yin, L. Qiu, Y. Liu, X. Tong, and X. Han. Sampro3d: Locating sam prompts in 3d for zero-shot scene segmentation, 2023.
- [42] K. Yadav, R. Ramrakhya, S. K. Ramakrishnan, T. Gervet, J. Turner, A. Gokaslan, N. Maestre, A. X. Chang, D. Batra, M. Savva, et al. Habitat-matterport 3d semantics dataset. *arXiv preprint arXiv:2210.05633*, 2022.
- [43] Y. Yang, X. Wu, T. He, H. Zhao, and X. Liu. Sam3d: Segment anything in 3d scenes, 2023.
- [44] Y. Yin, Y. Liu, Y. Xiao, D. Cohen-Or, J. Huang, and B. Chen. Sai3d: Segment any instance in 3d scenes, 2024.

- [45] Y. Zhao, W. Lv, S. Xu, J. Wei, G. Wang, Q. Dang, Y. Liu, and J. Chen. Detrs beat yolos on real-time object detection, 2023.

7. Contributions And Acknowledgements

We are thankful for Cem Gokmen’s advice and guidance on the whole project; CS231N teaching staff’s great support.

Tatiana Veremeenko designed and implemented the first part of the pipeline. Mu-sheng Lin prepared the 3D segmentation masks and 3D mesh rendering (some pilot studies on text-to-3D senses and 3D Gaussian Splatting). Addison Jadwin designed the object/orientation matching via CLIP and GPT and replacing the object in the original scene with the BEHAVIOR-1k object.

A. Appendix

A.1. 3D Scenes from Text Prompts

In recent research, efforts are made to extend the text-to-2D images to text-to-3D scenes. Examples of 3D meshes of text2room [10] are shown in Fig. 13 and Fig. 14, for the bathroom and living respectively. The corresponding meshes are demonstrated in Fig. 17 and Fig. 18, where the challenge parts are particularly the ceilings, the floors and the walls with holes and fragments. 3D Gaussian Splatting (3D GS) for 3D scenes becomes highly interesting topics due to its real-time performance and high quality images [11]. We conducted some pilot study from [6] shown in Fig. 19 with the detailed GPT4-O text description prompt.

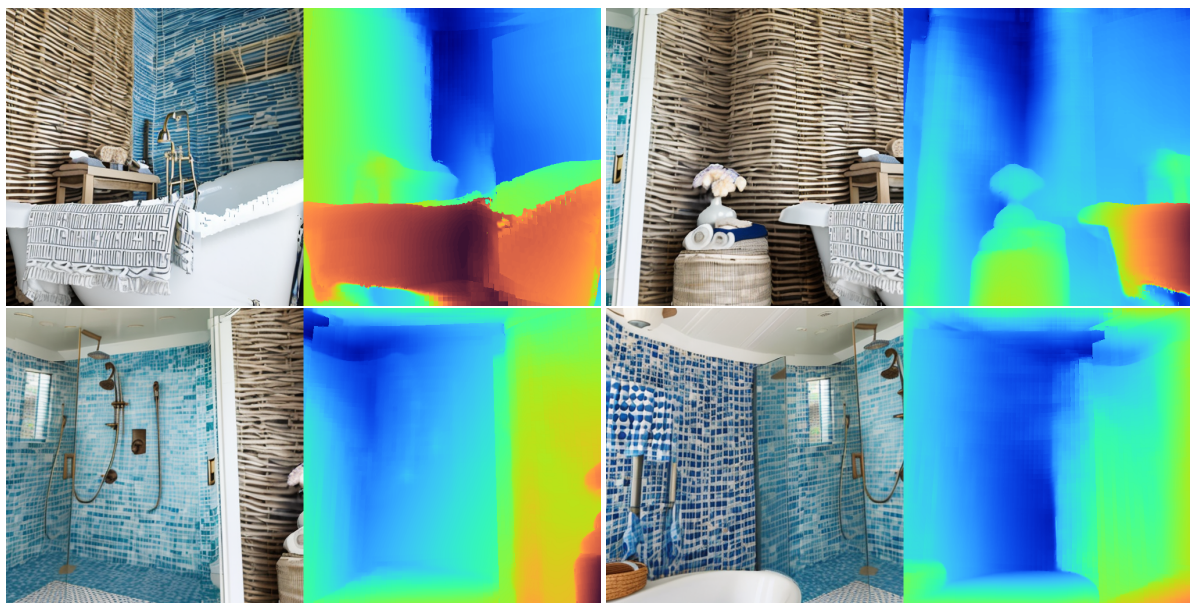


Figure 13: Example of bathroom scene.

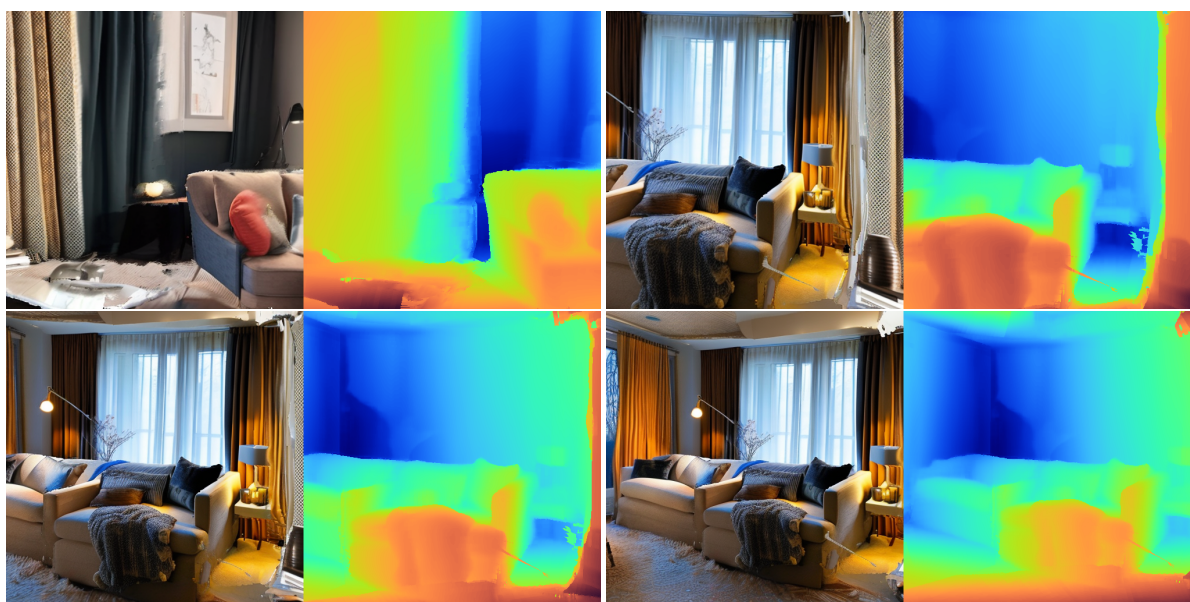


Figure 14: Example of living room scene.

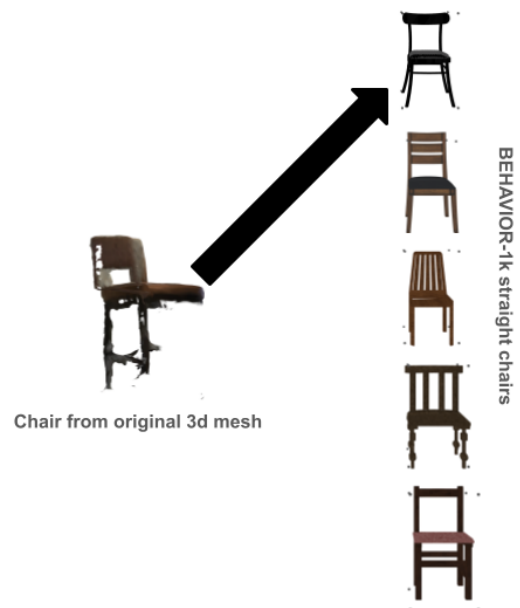


Figure 15: CLIP semantically matches the 3d mesh object to one of the objects in the dataset.

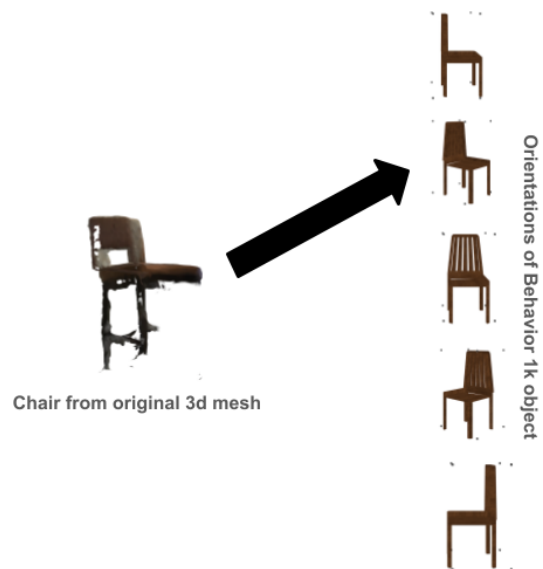


Figure 16: CLIP (with PCA) matches the 3d mesh object to one of the orientations of the already selected dataset object.



Figure 17: 3D Mesh of the Bathroom Scene.



Figure 18: 3D Mesh of the Living Room Scene.



Figure 19: 3D Gaussian Splatting with Depth Inpainting with detailed GPT4-O text description prompt.

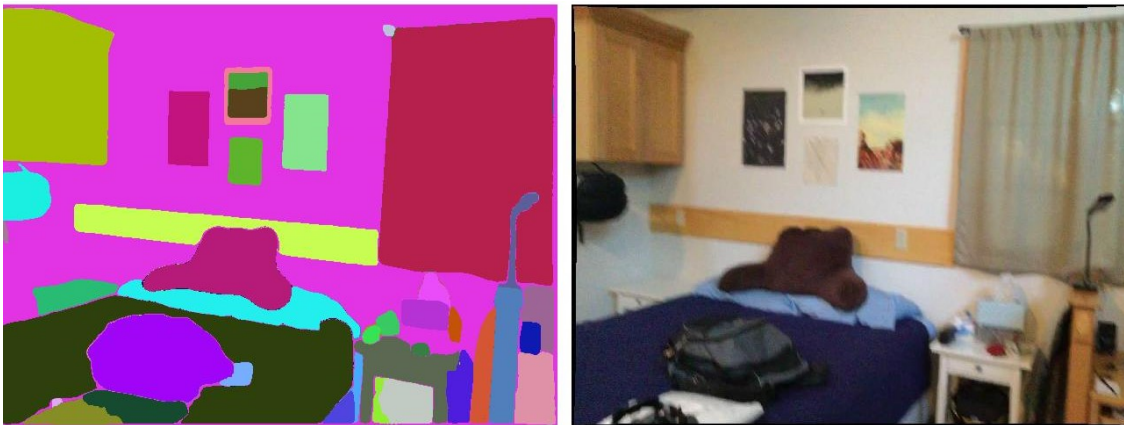


Figure 20: 2D segmentation results using SAM

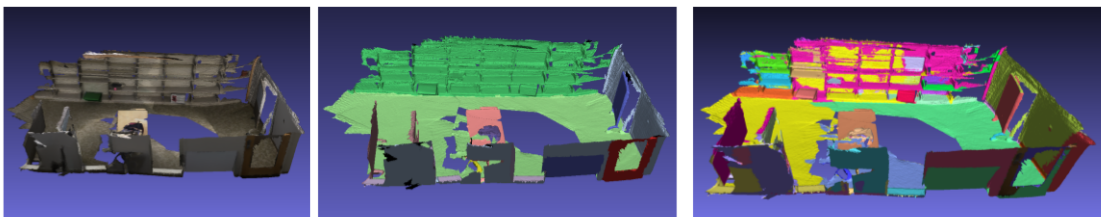


Figure 21: 3D segmentation results using SAMPro3D