Find Any Part in 3D

Ziqi Ma Yisong Yue Georgia Gkioxari
California Institute of Technology
{ziqima, yyue, georgia}@caltech.edu

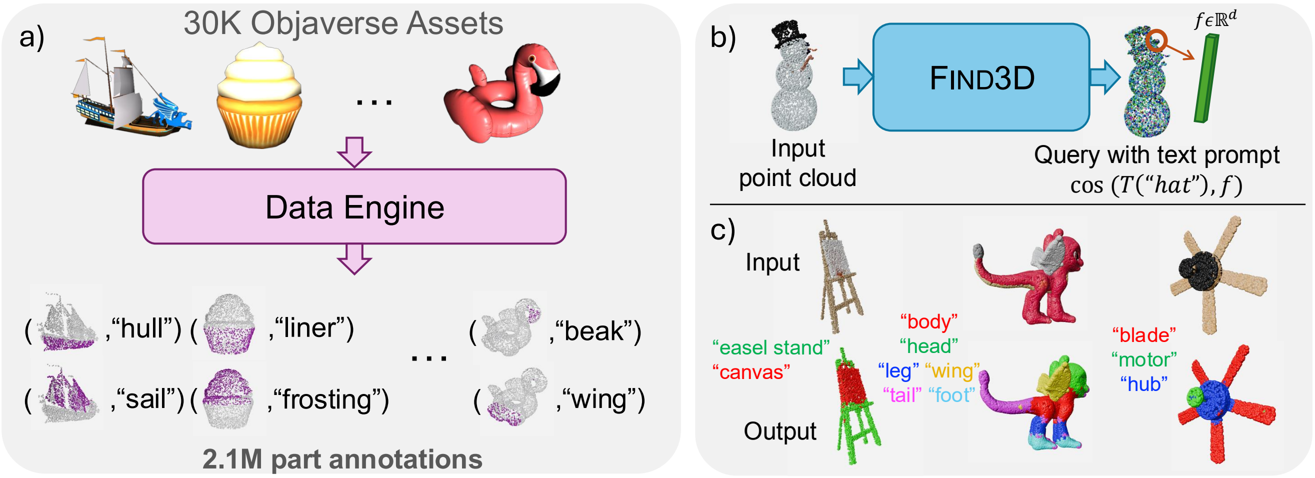

Figure 1. FIND3D is the first general-category 3D model that can segment *any* part of *any* object with *any* text query. We achieve this by building a scalable **Data Engine** powered by 2D foundation models – SAM & Gemini – that automatically annotates 3D assets from the web. Using the labeled data, **FIND3D** trains a transformer-based point cloud model with a contrastive training recipe. Our method works on diverse 3D objects and parts, *e.g*. the easel, the imaginary animal, and the ceiling fan.

Abstract

*Why don't we have foundation models in 3D yet? A key limitation is data scarcity. For 3D object part segmentation, existing datasets are small in size and lack diversity. We show that it is possible to break this data barrier by building a data engine powered by 2D foundation models. Our data engine automatically annotates any number of object parts: **1,755**× more unique part types than existing datasets combined. By training on our annotated data with a simple contrastive objective, we obtain an open-world model that generalizes to **any** part in **any** object based on **any** text query. Even when evaluated zero-shot, we outperform existing methods on the datasets they train on. We achieve* **260%** ***improvement in mIoU*** *and boost speed by **6**× to **300**×. Our scaling analysis confirms that this generalization stems from the data scale, which underscores the impact of our data engine. Finally, to advance general-category open-world 3D part segmentation, we release a benchmark covering a wide range of objects and parts. Project website:* https://ziqi-ma.github.io/find3dsite/

1. Introduction

Is it possible to build foundation models in 3D? For text and image modalities, we have seen that strong, general models come from internet-scale training data. In the absence of such large-scale 3D datasets, can we attempt to replicate this success in 3D?

In this paper, we provide an answer to this question. We show that when you tackle the data challenge, you can get a strong model with a simple, general training recipe. This approach not only unlocks generalization to unseen objects with a **260% improvement in mIoU**, but even outperforms prior methods on the datasets they train on.

Prior works which rely heavily on dataset-specific customization suffer from poor generalization because existing datasets are small and homogeneous. For example, ShapeNet-Part [27] contains only 16 categories and makes assumptions such as "all chairs face right". Evaluation on such limited datasets implicitly encourages dataset-specific customization, which is not the path towards generalization. Moreover, many prior works use pipelines that are computationally expensive and cannot scale to larger training sets.

In this paper, our goal is to: 1) establish a scalable data engine that can generate useful labels for any number of 3D assets; and 2) show that having a large-scale training set enables strong generalization with a simple training recipe, without any customizations for specific datasets such as per-category prompt and viewpoint search [33], category-specific finetuning [21, 32], multi-pass inference customization with predefined part ranking logic [32], or slow test-time pipelines [20]. Conceptually, our findings mirror those in other domains such as text, where using general training recipes at scale leads to powerful and general foundation models.

Concretely, as shown in Fig. 1, we enable scaling in 3D by building a data engine that automatically annotates synthetic 3D assets on the internet, yielding **2.1 million** part annotations of 761 object categories. Our dataset contains $124,615$ unique part types, which is over $\mathbf{1,775}\times$ the number of unique part types in existing datasets combined (ShapeNet-Part [27] and PartNet-E [12] contain 71 unique part types combined). To leverage such large-scale data, we devise a contrastive training objective to handle part hierarchy and ambiguity. Our model takes in a point cloud and predicts a queryable semantic feature for every point. The features are in the latent embedding space of a CLIP-like [15] model, so that they can be queried with any free-form text by calculating pointwise cosine similarities with the query embedding.

This approach yields a model that can segment *any* part of *any* object, with *any* text query. We highlight the following contributions:

- We develop a data engine that labels 3D object parts from large-scale internet data to train a general-category model **without the need for human annotation**. Our data engine creatively combines existing vision and language foundation models.
- We build the first model for 3D segmentation that is simultaneously **open-world**, **cross-category**, **part-level** and **feed-forward**. We achieve **260% improvement in mIoU** and **6**× to over **300**× the inference speed compared to existing methods.
- We release a **benchmark** for evaluating open-world 3D part segmentation for diverse objects, with **5**× more unique part types than the largest existing benchmark.

2. Related Work

Closed-world 3D segmentation. 3D segmentation has been studied primarily in a closed world and with a coarse granularity that cannot go below whole objects. In specific settings such as indoor scenes or self-driving, state-of-the-art models are starting to achieve better generalization by training on multiple datasets, such as Mask3D [18] and the PointTransformer series [22, 23, 30]. However, these models are still domain-specific, and can only segment whole objects rather than parts. Part-level segmentation is less studied. Early efforts started with the ShapeNet-Part dataset [27] (16 object classes, $\leq$ 6 parts per object). PartNet-E introduces articulated objects but is still limited to only 45 categories. Due to the limited number of categories and shared orientations (*e.g.*, chairs all facing right), state-of-the-art part-level models [9, 14] cannot generalize well. Our work tackles both the challenges of generalization and granularity – our model is part-level, and can segment any object part in an open-world setting.

3D aggregation methods based on 2D renderings. With the progress of vision language models in 2D image understanding, some works directly assemble these models to obtain an "aggregated" 3D understanding *without* training a 3D model. An exemplary aggregation method uses multiview renderings of 3D scenes or objects, obtains their features in 2D based on models like CLIP [15], SAM [6], or GLIP [7], and combines them in 3D based on projection geometry. On the whole object level, such methods include OpenMask3D [20]. On the part level, such methods include PointCLIP [29], PointCLIPV2 [33], PartSLIP [8] and PartSLIP++ [32] for point clouds, and SATR [1] for meshes. These models lack 3D geometry information and suffer from inconsistency across views. Furthermore, these methods are slow because they perform many inferences and the aggregation logic at test time. Our method, which predicts in 3D with a single inference, is significantly faster. Our method also achieves stronger performance and better robustness to pose changes by leveraging 3D geometry information.

Test-time optimization. Test-time optimization methods combine features from 2D models with a 3D representation, such as NeRF or Gaussian Splatting. At test time, these methods optimize the 3D representation with the 2D-sourced features attached. LERF [3], Distilled Feature Field [19], and Garfield [4] are based on radiance fields. Feature3DGS [31] is based on Gaussian splatting. These methods need to be optimized *per scene* (or *per object*), which can be slow (several minutes). Moreoever, their part-level capabilities have not been well-studied. Our method, feed-forward in nature, provides much faster inference with better performance.

Distillation methods. Distillation methods train 3D models using 2D annotations. Generalization is a key limitation in prior works – distillation is usually performed per dataset, even per category. OpenScene [13], a whole-object segmentation model for indoor scenes, is distilled per dataset. For part segmentation, PartDistill [21] is distilled *per category*. Such models cannot perform inference zero-shot on unseen object classes, which is critical in real-world use cases. Our approach can be considered a distillation method that tackles the challenge of zero-shot generalization.

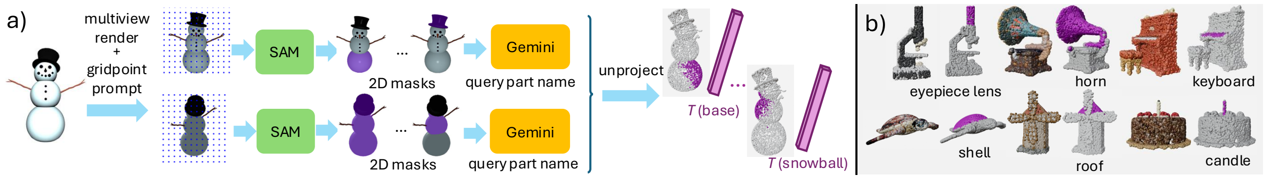

Figure 2. The **Data Engine**. (a) We render Objaverse assets into multiple views and pass each rendering to SAM with gridpoint prompts for segmentation. For each mask, we query Gemini for the corresponding part name, which gives us (mask, text) pairs. We embed the part name into the latent embedding space of a vision and language foundation model such as SigLIP. We back-project mask pixels to obtain the points associated with each label embedding, yielding (points, text embedding) pairs. (b) Example annotations by the data engine.

3. Method

We propose a method, FIND3D, to locate any object part in 3D based on a free-form language description, such as "the wheel of a car". As shown in Fig. 1 (panel b), we design a model that takes in a point cloud and outputs a queryable semantic feature for every point. This semantic feature is in the latent embedding space of a pre-trained CLIP-like [15] model, such as SigLIP [28]. For any text query, we embed the query using the same model and calculate its cosine similarity with each point's feature. This yields a pointwise similarity score that reflects the confidence of the part being located at that point. This score can be used to segment the object or localize specific parts.

Formally, given a point cloud $C = \{\mathbf{p_1}, ...\mathbf{p_n}\}$ with color and normals, for any point $\mathbf{p_i} = (x, y, z, n_x, n_y, n_z, r, g, b) \in \mathbb{R}^9$, we want to find a semantic feature $f_i \in \mathbb{R}^d$ which belongs in the same latent embedding space as a CLIP-like model, *e.g.*, SigLIP. At inference time, for any text s, we can get its SigLIP embedding $T(s)$ and compute its cosine similarity with f_i, $cos(T(s), f_i)$. For segmentation, FIND3D assigns each point to the text query with the highest cosine similarity, and assigns "no label" if all queries yield negative similarity scores.

3.1. Data Engine

Obtaining large-scale 3D annotations for generic object categories with human-in-the-loop pipelines is onerous. We develop a scalable data engine that leverages annotations from 2D foundation models and geometrically unprojects them to 3D.

As illustrated in Fig. 2, our data engine leverages SAM [6] and Gemini [17] to annotate 3D assets from Objaverse [2]. Since Objaverse assets do not have a fixed orientation, and Gemini provides higher-quality labels to objects seen in familiar orientations, we first prompt Gemini to select the best orientation based on 10 renderings (from different camera angles) of an object in each orientation. For the chosen orientation, we pass all renderings to SAM with grid point prompts. We discard masks that are too small (less than 350 pixels out of a 500×500 image), too large (greater than 20% of all pixels), or with low confidence from SAM. We overlay each mask on the original image and ask Gemini to name the shaded part. Prompts are detailed in the appendix. Masks with the same label are merged. This process generates labeled (mask, text) pairs. We map each mask to a set of points in the point cloud based on projection geometry. To make the point features queryable by language, we align point features to the language embedding space of a pretrained model, such as SigLIP. We embed the label texts and use the text features as supervision.

The data engine processes 36,044 Objaverse objects under LVIS categories selected by [10, 11]. Each part can be annotated differently from different views, denoting various aspects of part, such as location (*e.g.*, "bottom"), material (*e.g.*, "snowball"), and function (*e.g.*, "body"). Labels also have different levels of granularity. For example, in Fig. 2, one granularity is individual snowballs, and another granularity is the whole snowman. The diversity of our labels helps the model handle the inherent ambiguity in segmentation. Our data engine annotates 30K objects from 761 unique categories with 2.1 million parts in total. Our annotations contain 124,615 unique part types, which is over **1,775**× the number of unique part types in existing datasets combined (ShapeNet-Part and PartNet-E contain 71 unique part types in total). Fig. 2 panel b shows some example annotations covering a wide range of part types and object geometries. We provide more annotation examples by our data engine in the appendix.

3.2. Open-World 3D Part Model

Architecture. FIND3D adopts the PT3 [22] architecture that treats point clouds as sequences, as illustrated in Fig. 3. To align the point features into the latent embedding space of SigLIP, we append a lightweight 4-layer MLP to the last layer of the transformer. This returns a 768-dimension feature per point. Our model contains 46.2 million parameters.

Training. Leveraging the diverse annotations from our data engine requires some care. We cannot define a direct pointwise loss because: 1) The same point can have multiple labels that denote various aspects of a part such as location, material, and function. Some labels may also be incor-

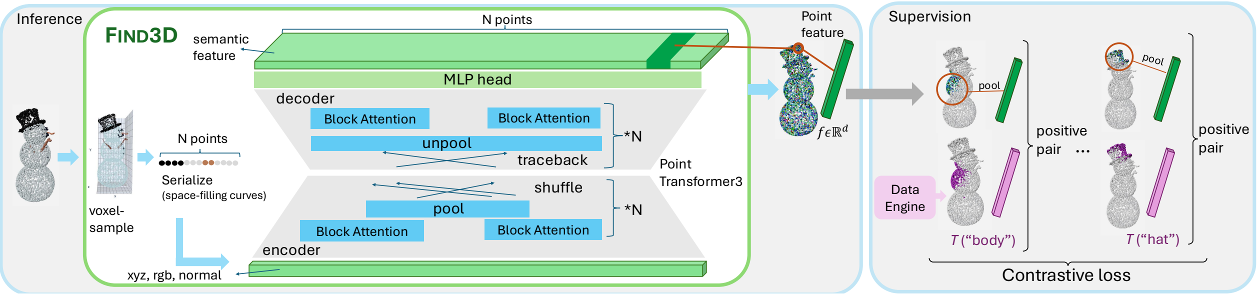

Figure 3. **FIND3D**: an open-world part segmentation model. FIND3D takes in a point cloud, voxelizes and serializes the points via space-filling curves into a sequence. The sequence is passed through a transformer architecture which returns a pointwise feature that is in the embedding space of a vision and language foundation model, denoted by T. These features can be queried with any free-form text. FIND3D is trained with a contrastive objective. For each (points, text embedding) label from the data engine, we use the averaged feature of these points as the predicted embedding, and pair it with the text embedding to form a positive pair in the contrastive loss.

rect; and 2) Many points are unlabeled - as shown in Fig. 3 (right), each mask only labels points visible from one camera view, and thus parts are likely to be labeled partially.

The challenge of partial labels can be resolved if the model can map features based on 3D geometry: in the snowman example of Fig. 3, points on the same snowball should share similar features, and if we align the features of some points on that ball correctly to the text embedding, the other points' features should also be aligned. The challenge of multiple labels can be resolved by the contrastive formulation: each point's feature is encouraged to be close to the embeddings of all its labels, which allows for flexible text queries at inference time. As illustrated in Fig. 3 (right), we define the contrastive pairing as follows: for each label, the ground truth is the SigLIP embedding of the text. The predicted value is the average feature of all points that correspond to the label. This pooling can also be regarded as a way to "denoise" the labels – while an individual point might be affected by conflicting or incorrect labels, it is unlikely that all points are subjected to the same error.

Formally, our data engine provides (points, text embedding) labels, which we denote as $(C_i, T(\text{label}_i))$ where C_i is a subset of the point cloud that this label corresponds to, and $T(\text{label}_i)$ is the label embedding. We denote the pooled feature from the labeled points as $f(C_i)$, where f is our model. We define the contrastive loss as follows:

$$l_i = -\log \frac{\exp(f(C_i) \cdot T(\text{label}_i))}{\sum_{j=1}^{|\mathcal{B}|} \exp(f(C_i) \cdot T(\text{label}_j))} \tag{1}$$

where $\mathcal{B}$ denotes all labels of all objects in a batch. For training, we use a batch size of 64 objects, corresponding to $\sim 3,000$ positive pairs per batch.

To achieve generalization, in addition to training on diverse data provided by the data engine, we also apply data augmentations, including random rotation (implemented as sequential random rotation along all three axes), scaling, flipping, jittering, chromatic auto contrast, chromatic translation, and chromatic jitter. These augmentations help avoid over-reliance on object poses and color, and nudges the model to take up 3D geometric cues. We perform a 90:10 train-validation split on the 27,552 objects provided by the data engine, and train with the Adam optimizer [5] with a cosine annealing learning rate schedule, starting at 0.0003 and ending at 0.00005 over 80 epochs.

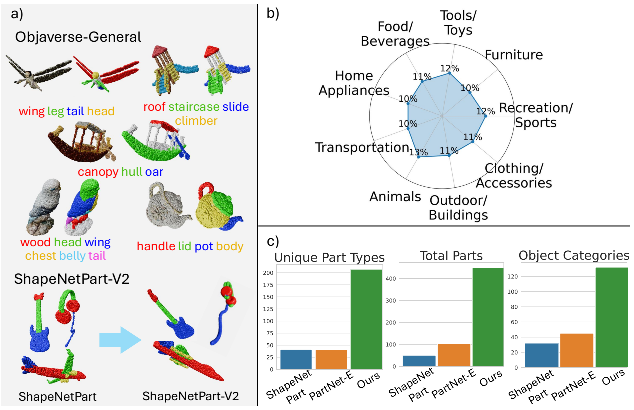

Figure 4. Our benchmark. (a) Examples of Objaverse-General and ShapeNetPart-V2. Objaverse-General contains diverse objects and parts, and ShapeNetPart-V2 is sourced to look similar to ShapeNet-Part to test various methods' generalization capability. (b) Object category breakdown of Objaverse-General, which covers 9 categories from tools to buildings. (c) Comparison with existing benchmarks. We have **5**× more unique part types, **4.4**× more total annotated parts, and **2.9**× more object categories.

4. A General Open-World 3D Part Benchmark

Existing 3D part segmentation benchmarks only contain a small number of categories with a fixed set of parts, and are limited to narrow domains. ShapeNet-Part [27] contains 16 object categories with 41 unique part types, and the domain is limited to CAD models with canonical orientations (*e.g.*, all chairs face right). PartNet-E [12, 24] contains 45 cate-

Method	Time	Open-world	Cross-category	Part-level	Feed-forward
FIND3D	0.9s	✓	✓	✓	✓
PointCLIPV2	5.4s	✓	✓	✓	✗
PartSLIP++	174.3s	✓	✗*	✓	✗
OpenMask3D	296.5s	✓	✓	✗	✗
PartDistill†	0.7s (+348s)	✗	✗	✓	✓
PointNext	1.4s	✗	✓	✓	✓

Table 1. Properties and inference time of FIND3D and baselines. FIND3D is the only method that is open-world, cross-category, part-level and feed-forward. By "feed-forward", we mean a method that performs direct 3D inference, without relying on a pipeline of multiview rendering, multiple 2D model inferences, and backprojection. Our method is $\mathbf{6}\times$ to $\mathbf{300}\times$ faster than other open-world models, and on par with closed-world models. $*$ PartSLIP++ finetunes a model per category. † PartDistill performs distillation for each new category (348s) and then does inference (0.7s). It is not open-world as the part names need to be defined prior to distillation. It only releases source code for two categories, and our reported speed is averaged across them.

gories with 40 unique part types (*e.g.*, "button" is a common part across categories), and is restricted to simple home objects such as bottles and doors. As shown in Fig. 4, we introduce a new human-annotated benchmark featuring a diverse range of objects, shapes, parts, and poses. We source our data from Objaverse [2]. Our benchmark contains 132 object categories, 450 total parts and 207 unique part types, over $\mathbf{5}\times$ that of existing benchmarks, as shown in Fig. 4. The annotation protocol is detailed in the appendix. We hope this benchmark can advance 3D part segmentation towards more variable, "in-the-wild" scenarios. The benchmark is divided into two sets:

Objaverse-General contains 100 objects with 350 parts from 100 diverse object categories, such as gondola, slide, lamppost, easel, penguin. These objects are in random orientations. We hold out 50 out of the 100 categories from training, in order to evaluate out-of-distribution generalization to novel object types. The holdout categories are termed Unseen-Categories in Table 2.

ShapeNetPart-V2 contains 32 objects from the same 16 object categories in ShapeNet-Part [27]. Inspired by ImageNetV2 [16], we create this benchmark to evaluate generalization for models trained on ShapeNet-Part.

5. Experiments

As summarized in Table 1, FIND3D is the first method that is simultaneously, open-world, cross-category, part-level and feed-forward. FIND3D not only shows strong zero-shot generalization, but also outperforms existing methods on their own domain. Our experiments show:

- FIND3D achieves strong performance on diverse objects, with **260% improvement in mIoU** from existing methods. FIND3D exhibits strong out-of-distribution generalization, whereas baseline methods perform poorly on datasets they are not trained on, as shown qualitatively in Fig. 5 and quantitatively in Tab. 2, Tab. 3, Tab. 4.
- FIND3D is robust to variations such as query prompt rephrasing, object rotation, and domain shift, whereas baselines are sensitive to these changes. This is shown in Fig. 7, Tab. 3, Tab. 4.
- FIND3D is the most efficient open-world method with **6x** to **300x** speed improvement, as shown in Tab. 1.

5.1. Experimental Settings

Benchmarks. In addition to our proposed benchmark (Sec. 4), we also evaluate on two commonly used datasets for 3D part segmentation: ShapeNet-Part [27] (16 object categories) and PartNet-E [12] (45 object categories). For both datasets, we evaluate on their test set both in the canonical pose and in a randomly rotated (around all axes) pose, which correspond to the *Canonical* and *Rotated* columns in Tab. 2, Tab. 3.

Metric. We report class-average intersection-over-union (mIoU) as our metric, which is the mean IoU for all labeled parts per object, averaged across all object categories.

Competing Methods

Open-world Baselines: **PointCLIPV2** [33] is an open-world 2D-to-3D pipeline involving multiple invocations of CLIP [15]. It uses top-k prompts ($k = 1400 \times n_{\text{parts}}$ per object) selected on the test set of ShapeNet-Part. **PartSLIP++** [32] is a detection-based pipeline involving invocations of GLIP [7] and a custom algorithm for finding superpoints. It finetunes a separate model for each category in PartNet-E. We evaluate its zero-shot checkpoint for fairness of comparison. **OpenMask3D** [20] is an open-vocabulary, 2D-to-3D pipeline trained on scenes.

PointCLIPV2 and OpenMask3D are dense methods that assign a label to every point. We provide the text query "other" as an option for no label on benchmarks that contain unlabeled points.

Closed-world Baselines: **PointNeXt** [14] is a state-of-the-art closed-world point cloud segmentation model trained on ShapeNet-Part. Due to its closed vocabulary, it cannot be evaluated on other datasets. **PartDistill** [21] is a category-specific 2D-to-3D distillation method, which is open-world prior to distillation but closed-world at inference time. It cannot be evaluated on unseen object categories due to the category-specific nature of distillation. The code and data for this method are not fully released (only two categories are released). Since we cannot reproduce the approach, we show numbers claimed in the paper.

Because PartSLIP++ and OpenMask3D are slow (up to 5 minutes per object), they are infeasible to evaluate on the full test sets (evaluating OpenMask3D on PartNet-E test set would take 628 hours). For fair evaluations of all methods, we create smaller subsets of 160 objects (10 objects/category × 16 categories) for ShapeNet-Part and 225

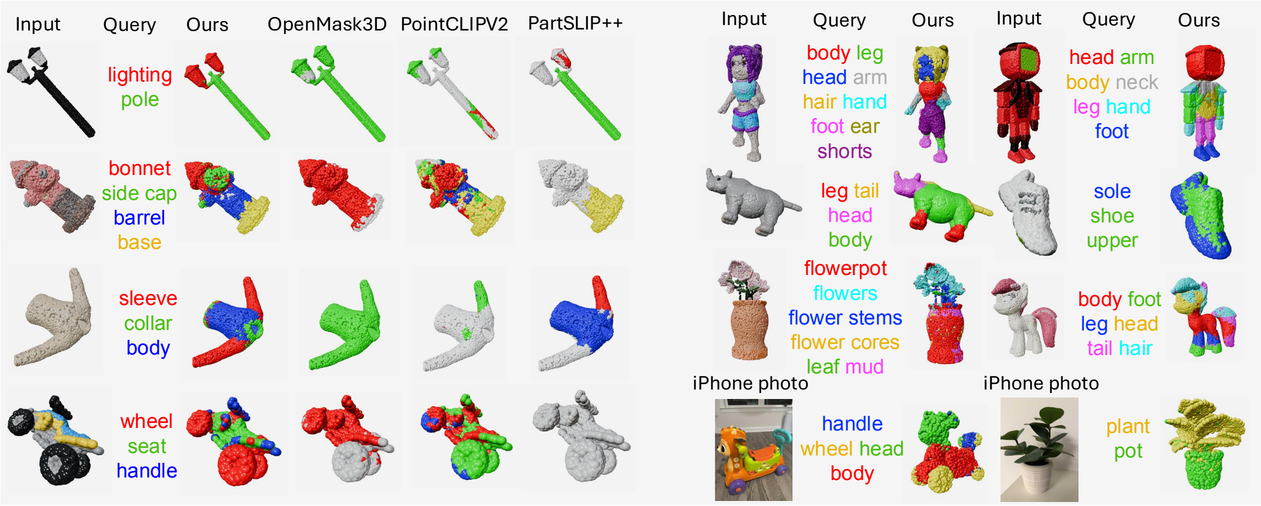

Figure 5. Qualitative results. Left: FIND3D performs strongly on Objaverse-General while baseline methods struggle. Right: more examples both from Objaverse-General and PartObjaverse-Tiny, including out-of-distribution objects such as magical animals and complex anime-style characters. FIND3D works on diverse object categories with up to 9 parts. It also generalizes to "in-the-wild" iPhone photos (converted to point clouds via off-the-shelf image-to-3D method, as shown at bottom right.

mIoU (%)	Objaverse-General				ShapeNet-Part					
	Seen Categories		Unseen Categories		Canonical		Rotated		ShapeNetPart-V2	
	{part} of a {object}	{part}	{part} of a {object}	{part}	{part} of a {object}	{part}	{part} of a {object}	{part}	{part} of a {object}	{part}
FIND3D (ours)	**33.78**	**34.10**	**26.21**	**27.41**	**28.39**	**24.09**	**29.64**	**23.71**	**42.15**	**30.02**
PointCLIPV2	9.81	11.27	10.27	11.09	16.91	20.22	16.88	18.19	15.14	17.11
PartSLIP++	2.69	15.03	0.57	10.43	1.43	6.46	0.94	6.03	1.54	11.62
OpenMask3D	11.81	11.93	7.01	10.31	8.94	10.37	6.75	14.56	15.87	13.77

Table 2. Performance comparison with open-world methods on Objaverse-General and ShapeNet-Part. Shaded cells mean the method is trained on the same dataset (expected higher than white cells); white cells mean zero-shot evaluation. FIND3D performs best on Objaverse-General, with **260% improvement in mIoU** on unseen categories where all methods are evaluated zero-shot. On ShapeNet-Part, FIND3D's zero-shot performance even exceeds PointCLIPV2 which is trained on the this dataset. We show results evaluated with 2 common query prompts: "{part} of a {object}" and "{part}" for all methods.

objects (5 objects/category × 45 categories) for PartNet-E. For methods that are efficient to evaluate, we additionally report performance on the full test sets in the appendix. We observe the same rankings and similar results on the subsets and full sets.

5.2. Experimental Results

Results on Objaverse-General and ShapeNet-Part. Tab. 2 reports the mIoU of FIND3D and open-world baselines on Objaverse-General and ShapeNet-Part. FIND3D shows the strongest performance, with **260% improvement in mIoU** compared to the best baseline, PointCLIPV2, when both are evaluated zero-shot out of distribution (Objaverse-General– Unseen Categories). Additionally, even when evaluated zero-shot, FIND3D outperforms PointCLIPV2 on ShapeNet-Part, the dataset it is trained on.

Qualitative results. As seen in Fig. 5, FIND3D consistently outputs reasonable segmentations, while other methods struggle. PartSLIP++ is trained on PartNet-E with sparse part annotations, and thus tends to output "no label" overly often. OpenMask3D struggles with the part-level granularity, and it usually only picks one or two parts to represent the whole object. We additionally show examples in Objaverse-General and PartObjaverse-Tiny [26] for our method on the right. Notably, FIND3D can segment parts that are not easily segmentable in 2D due to the lack of visible edges, such as the sleeves of a monochromatic shirt. FIND3D not only works with diverse objects and parts, but also generalizes to real-world iPhone photos (converted to point clouds with Trellis [25], an off-the-shelf image-to-3D model), despite only being trained on synthetic assets.

Results on PartNet-E. Tab. 3 compares open-world methods on PartNet-E. PartSLIP++ is trained on this dataset while all other methods, including FIND3D, are evaluated zero-shot, so the results in this table favor PartSLIP++. We evaluate on two common prompts: "part of a object" (such as "leg of a chair"), and "part name" ("leg"). We see that PartSLIP++'s performance decreases greatly when we vary the query prompt, up to a 83% drop. Our method, evaluated zero-shot, is more robust and outperforms PartSLIP++

mIoU(%)	Canonical Orientation		Rotated	
	{part} of a {object}	{part}	{part} of a {object}	{part}
FIND3D (ours)	**16.86**	16.38	**17.62**	17.16
PartSLIP++	5.12	**32.71**	3.87	**23.03**
PointCLIPV2	11.28	9.70	10.32	10.22
OpenMask3D	12.54	11.24	11.93	11.67

Table 3. Comparison of open-world methods on PartNet-E. Shaded cells mean the method is trained on the same dataset (expected higher than white cells); white cells mean zero-shot evaluation. We evaluate with 2 prompt formats: "{part} of a {object}" and "{part}". PartSLIP++ achieves good performance with the "{part}" prompt, but its performance drops **84**% when we vary the query prompt. This dataset is challenging for our method due to the sparsity of labels and the presence of small parts that are not geometrically or colorfully prominent (*e.g.*, buttons on a surface with the same color). Nevertheless, our method is more robust to rotation and prompt variation, and clearly outperforms the other baselines that are also evaluated zero-shot.

with the "part of a object" prompt. It also outperforms other zero-shot baselines under all evaluation configurations.

Efficiency. As shown in Tab. 1, FIND3D only takes 0.9 seconds for inference, which is **6**× to **300**× faster than open-world baselines and on par with closed-world models. Inference time is the average per-object inference time on the PartNet-E subset evaluated on an A100.

Comparing with closed-world methods on ShapeNet-Part and ShapeNetPart-V2. Tab. 4 compares FIND3D zero-shot with closed-world methods that are trained on ShapeNet-Part, which greatly favors the closed-world methods. PointNeXt is the leading closed-world method for this dataset, and PartDistill trains one model for each object category of ShapeNet-Part. We additionally evaluate the methods' generalization capability on our ShapeNetPart-V2 benchmark, similar to ImageNetV2 [16]. We see a **64% drop** of PointNeXt. Even though PointNeXt is still in-domain and FIND3D is evaluated out-of-distribution, FIND3D shows a **1.5**× advantage. PartDistill is not reproducible and thus cannot be evaluated on ShapeNetPart-V2.

5.2.1. Scaling Analysis

Data scaling is critical, as shown by the scaling analysis in Fig. 6. This finding highlights the importance of our data engine approach, which enables scaling in 3D. We vary training object categories (x-axis) ranging from 16 categories (ShapeNet-Part dataset size), 45 categories (PartNet-E dataset size), all the way to 761 (our setting). We report zero-shot mIoU on Objaverse-Unseen Categories (y-axis). We observe a strong scaling trend which is consistent with findings in many other data domains.

5.2.2. Quantifying and Comparing Robustness

Fig. 7 evaluates the robustness of our method under different query text prompt, object orientation, and data domain, i.e. the data source of similarly-looking objects. We qualitatively show our model's robustness to fine-grained parts and parts without edge delineation in the appendix.

mIoU (%)	Trained on	ShapeNet-Part	ShapeNetPart-V2
FIND3D	Our data engine	28.39	42.15
PointNeXt	ShapeNet-Part	80.44	28.70
PartDistill†	ShapeNet-Part	63.9	N/A

Table 4. Performance comparison with closed-world methods. Shaded cells mean the method is trained on the same dataset (expected higher than white cells); white cells mean zero-shot evaluation. PointNeXt, a state-of-the-art closed-world model, is trained on ShapeNet-Part, but its performance drops significantly on ShapeNetPart-V2. Our approach, which is trained on a domain different from either ShapeNet-Part and ShapeNetPart-V2, demonstrates a stronger out-of-domain performance (1.5× better on ShapeNetPart-V2, +13.2%). † PartDistill trains a model per-category on ShapeNet-Part. It does not release training source-code or checkpoints (apart from two categories), thus cannot be evaluated on ShapeNetPart-V2.

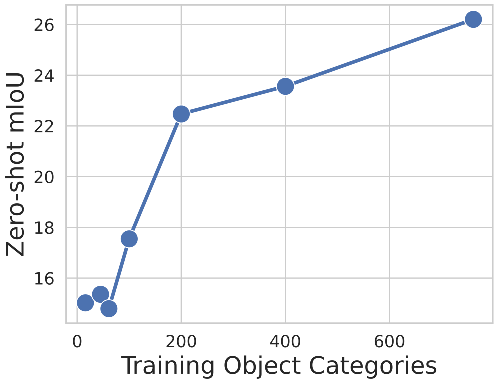

Figure 6. Data scaling which shows that training on more object categories provides clear improvement in zero-shot mIoU. The evaluation is done on Objaverse-General Unseen Categories.

Robustness to query prompt. PointCLIPV2 performs an extensive top-k prompt search on the test set: they iteratively optimize the prompt for each part (iteratively searching over 700 prompts per part and looping over all parts twice) and select the best prompt, i.e. iterate over $k = 1400 \times n_{\text{part}}$ prompts and pick the one with the best test performance. For a fair comparison, we perform the same top-k search for our method. We also evaluate on two common prompts: "part of a object" (such as "leg of a chair"), and "part name" ("leg"). As shown in Fig. 7, with a change of prompt from top-k to "part of a object", PointCLIPV2's performance drops from 48.47 to 17.42 (64% decrease), whereas our method exhibits more robust performance.

Robustness to object orientation. We apply a random rotation by sampling three angles from $-\pi$ to π and applying rotations along each of the X, Y, Z axis sequentially. PointCLIPV2's performance drops 46% whereas our

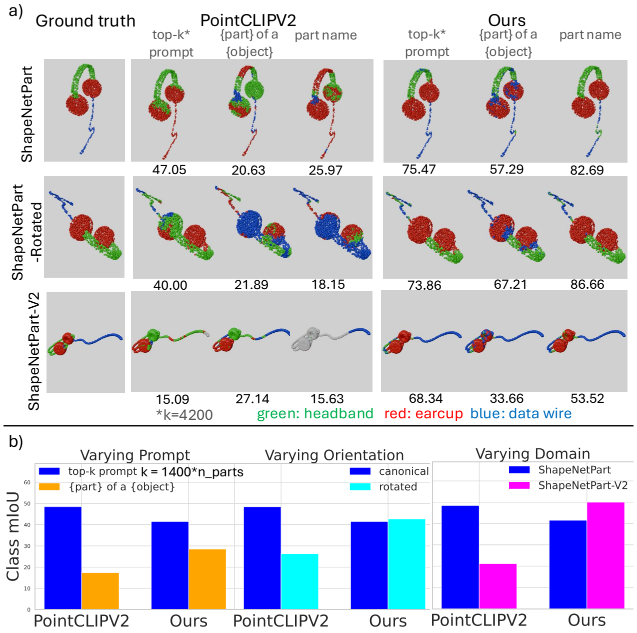

Figure 7. (a) Qualitative comparison of PointCLIPV2 and FIND3D on a ShapeNet-Part earphone (canonical and rotated) and a visually similar earphone from ShapeNetPart-V2. Top-k prompt reproduces evaluation in the PointCLIPV2 paper. PointCLIPV2's performance drops up to 68%, whereas our method stays consistent. (b) Comparison over all ShapeNet-Part categories. PointCLIPV2's performance drops 46% to 64% with varying conditions, while our method remains robust.

method does not drop but even increases 3%.

Robustness to domain. We constructed ShapeNetPart-V2, a benchmark with objects from the same categories as ShapeNet-Part, but sourced from Objaverse assets. With this domain shift, PointCLIPV2's performance drops from 48.47 to 21.18 by 56%, whereas our method stays robust with a 20% increase.

Comparison with other methods on other datasets in Tab. 2 and Tab. 3 show similar trends.

5.2.3. Flexibility of text queries

FIND3D supports various query types that might occur in-the-wild. As shown in Fig. 8, FIND3D can locate hands via different query types – either by the body part "hand" or by the clothing "gloves". The teddy bear example demonstrates flexibility in query granularity – one can query with "limbs", a combination of arms and legs, or with "arms" and "legs" separately. For ease of visualization, the scores are min-clipped at 0.

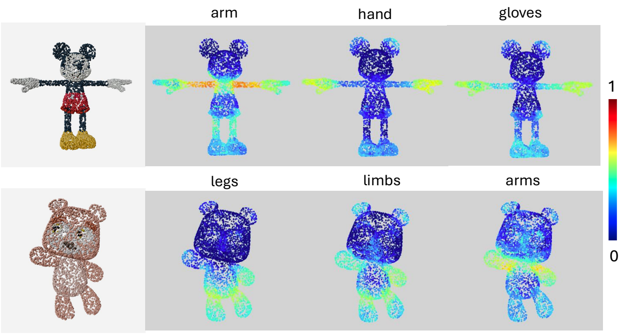

Figure 8. Our method can support flexible text queries. For Mickey, one can either query by a body part such as "hand" or by clothing such as "gloves". For the teddy bear, one can either query the coarser-granularity concept "limbs" or the finer-granularity "arms" and "legs".

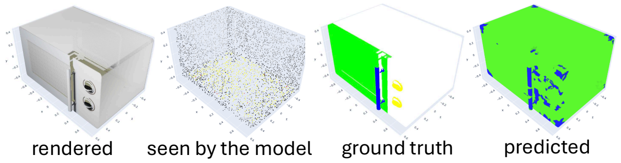

Figure 9. A failure example. The leftmost image is a rendering of a microwave. The second image shows the point cloud at FIND3D's sampled granularity, which loses most features.

5.2.4. Failure Modes

We observe some limitations of FIND3D: 1) Our model voxel-samples point clouds at the 0.02 resolution (after normalization). Fine-grained parts that are not geometrically prominent, such as bottons on a surface, are difficult for a point-cloud-only model like ours. 2) Because the model is trained to be rotational-equivariant, it tends to make symmetric predictions where all symmetric parts have the same label. Fig. 9 demonstrates an example from the PartNet-E dataset. These limitations point to the complementary nature of the 2D and 3D modalities. While lacking in 3D geometry, the 2D modalities can better convey detailed appearance. Combining the image and the point cloud modality is a future direction.

6. Discussions and Conclusions

We present the first scaling study for 3D part segmentation. Key to our approach is a data engine that automatically annotates 3D assets from the internet, which allows us to train the first zero-shot generalist model for open-world 3D part segmentation on *any* object. Our method not only shows strong generalization, but even outperforms prior methods on the datasets they train on, despite being zero-shot. We show that training object diversity is critical with a scaling analysis. We will release our code, benchmark and model checkpoints. We hope that by providing a diverse benchmark and the first demonstration of open-world 3D part segmentation at scale, we can encourage the community to shift away from customizations for small-scale datasets towards scale and generalization.

7. Acknowledgments

We would like to thank Ilona Demler, Raphi Kang, Damiano Marsili, Aadarsh Sahoo, and Jiacheng Liu for feedback on the paper draft. We also thank Jiacheng Liu for help with the project demo. Ziqi Ma is supported by the Kortschak scholarship. This project is funded in part by NSF #1918655, William H. Hurt Scholars Program, Powell Foundation, Google, and Amazon.

References

[1] Ahmed Abdelreheem, Ivan Skorokhodov, Maks Ovsjanikov, and Peter Wonka. Satr: Zero-shot semantic segmentation of 3d shapes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15166–15179, 2023. 2

[2] 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, 5

[3] Justin Kerr, Chung Min Kim, Ken Goldberg, Angjoo Kanazawa, and Matthew Tancik. Lerf: Language embedded radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19729–19739, 2023. 2

[4] Chung Min Kim, Mingxuan Wu, Justin Kerr, Ken Goldberg, Matthew Tancik, and Angjoo Kanazawa. Garfield: Group anything with radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21530–21539, 2024. 2

[5] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 4

[6] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4015–4026, 2023. 2, 3

[7] Liunian Harold Li, Pengchuan Zhang, Haotian Zhang, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, et al. Grounded language-image pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10965–10975, 2022. 2, 5

[8] Minghua Liu, Yinhao Zhu, Hong Cai, Shizhong Han, Zhan Ling, Fatih Porikli, and Hao Su. Partslip: Low-shot part segmentation for 3d point clouds via pretrained image-language models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 21736–21746, 2023. 2

[9] Marios Loizou, Siddhant Garg, Dmitry Petrov, Melinos Averkiou, and Evangelos Kalogerakis. Cross-shape attention for part segmentation of 3d point clouds. In *Computer Graphics Forum*, page e14909. Wiley Online Library, 2023. 2

[10] Xiaoxiao Long, Yuan-Chen Guo, Cheng Lin, Yuan Liu, Zhiyang Dou, Lingjie Liu, Yuexin Ma, Song-Hai Zhang, Marc Habermann, Christian Theobalt, et al. Wonder3d objaverse subset uids. `https://github.com/xxlong0/Wonder3D/blob/main/data_lists/lvis_uids_filter_by_vertex.json`. Accessed: 2024-11-01. 3

[11] Xiaoxiao Long, Yuan-Chen Guo, Cheng Lin, Yuan Liu, Zhiyang Dou, Lingjie Liu, Yuexin Ma, Song-Hai Zhang, Marc Habermann, Christian Theobalt, et al. Wonder3d: Single image to 3d using cross-domain diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9970–9980, 2024. 3

[12] Kaichun Mo, Shilin Zhu, Angel X Chang, Li Yi, Subarna Tripathi, Leonidas J Guibas, and Hao Su. Partnet: A large-scale benchmark for fine-grained and hierarchical part-level 3d object understanding. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 909–918, 2019. 2, 4, 5

[13] Songyou Peng, Kyle Genova, Chiyu Jiang, Andrea Tagliasacchi, Marc Pollefeys, Thomas Funkhouser, et al. Openscene: 3d scene understanding with open vocabularies. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 815–824, 2023. 2

[14] Guocheng Qian, Yuchen Li, Houwen Peng, Jinjie Mai, Hasan Abed Al Kader Hammoud, Mohamed Elhoseiny, and Bernard Ghanem. Pointnext: revisiting pointnet++ with improved training and scaling strategies. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, 2024. 2, 5

[15] 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, 3, 5

[16] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *International conference on machine learning*, pages 5389–5400. PMLR, 2019. 5, 7

[17] Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, Timothy Lillicrap, Jean-baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024. 3

[18] Jonas Schult, Francis Engelmann, Alexander Hermans, Or Litany, Siyu Tang, and Bastian Leibe. Mask3d: Mask transformer for 3d semantic instance segmentation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8216–8223. IEEE, 2023. 2

[19] William Shen, Ge Yang, Alan Yu, Jansen Wong, Leslie Pack Kaelbling, and Phillip Isola. Distilled feature fields enable few-shot language-guided manipulation. In *7th Annual Conference on Robot Learning*, 2023. 2

[20] Ayça Takmaz, Elisabetta Fedele, Robert W. Sumner, Marc Pollefeys, Federico Tombari, and Francis Engelmann.

OpenMask3D: Open-Vocabulary 3D Instance Segmentation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. 2, 5

[21] Ardian Umam, Cheng-Kun Yang, Min-Hung Chen, Jen-Hui Chuang, and Yen-Yu Lin. Partdistill: 3d shape part segmentation by vision-language model distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3470–3479, 2024. 2, 5

[22] Xiaoyang Wu, Li Jiang, Peng-Shuai Wang, Zhijian Liu, Xihui Liu, Yu Qiao, Wanli Ouyang, Tong He, and Hengshuang Zhao. Point transformer v3: Simpler faster stronger. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4840–4851, 2024. 2, 3

[23] Xiaoyang Wu, Yixing Lao, Li Jiang, Xihui Liu, and Hengshuang Zhao. Point transformer v2: grouped vector attention and partition-based pooling. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, 2024. 2

[24] Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, Li Yi, Angel X. Chang, Leonidas J. Guibas, and Hao Su. SAPIEN: A simulated part-based interactive environment. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 4

[25] 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. *arXiv preprint arXiv:2412.01506*, 2024. 6

[26] Yunhan Yang, Yukun Huang, Yuan-Chen Guo, Liangjun Lu, Xiaoyang Wu, Edmund Y. Lam, Yan-Pei Cao, and Xihui Liu. Sampart3d: Segment any part in 3d objects, 2024. 6

[27] Li Yi, Vladimir G Kim, Duygu Ceylan, I-Chao Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, and Leonidas Guibas. A scalable active framework for region annotation in 3d shape collections. *ACM Transactions on Graphics (ToG)*, 35(6):1–12, 2016. 1, 2, 4, 5

[28] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11975–11986, 2023. 3

[29] Renrui Zhang, Ziyu Guo, Wei Zhang, Kunchang Li, Xupeng Miao, Bin Cui, Yu Qiao, Peng Gao, and Hongsheng Li. Pointclip: Point cloud understanding by clip. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8552–8562, 2022. 2

[30] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 16259–16268, 2021. 2

[31] Shijie Zhou, Haoran Chang, Sicheng Jiang, Zhiwen Fan, Zehao Zhu, Dejia Xu, Pradyumna Chari, Suya You, Zhangyang Wang, and Achuta Kadambi. Feature 3dgs: Supercharging 3d gaussian splatting to enable distilled feature fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21676–21685, 2024. 2

[32] Yuchen Zhou, Jiayuan Gu, Xuanlin Li, Minghua Liu, Yunhao Fang, and Hao Su. Partslip++: Enhancing low-shot 3d part segmentation via multi-view instance segmentation and maximum likelihood estimation. *arXiv preprint arXiv:2312.03015*, 2023. 2, 5

[33] Xiangyang Zhu, Renrui Zhang, Bowei He, Ziyu Guo, Ziyao Zeng, Zipeng Qin, Shanghang Zhang, and Peng Gao. Pointclip v2: Prompting clip and gpt for powerful 3d open-world learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2639–2650, 2023. 2, 5