VLS: A Vision-Language-Shape Model for Open-Vocabulary Partonomic 3D Reconstruction

Xiaoqian Ruan
University of Illinois Chicago
xruan9@uic.edu

Pei Yu
Microsoft
pei.yu@microsoft.com

Dian Jia
University of Illinois Chicago
djia7@uic.edu

Hyeonjeong Park
University of Illinois Chicago
hpark233@uic.edu

Peixi Xiong
Intel
peixi.xiong@intel.com

Wei Tang
University of Illinois Chicago
tangw@uic.edu

Abstract

Single-view 3D reconstruction has advanced rapidly in recent years, but existing research primarily focuses on recovering whole-object geometry while ignoring their semantic parts, which are crucial for fine-grained 3D perception and downstream applications. This paper introduces the task of open-vocabulary partonomic reconstruction. Given a single image and a set of part names, we aim to reconstruct both the object's overall shape and its constituent semantic parts, even when the object and part categories are unseen during training. To address this task, we propose a vision-language-shape (VLS) model that unifies vision, language, and shape representations within a shared neural field and grounds continuous 3D coordinates to 2D pixel contexts via a deformable implicit function. It can effectively reconstruct any-topology shapes and open-vocabulary semantic parts from a single image. To train VLS with limited 3D part-labeled data, we propose an omni-supervised learning framework that leverages heterogeneous datasets with different levels of annotation and open-world knowledge from existing vision-language models. Extensive experiments on ShapeNetPart, PartNet, and Objaverse demonstrate the effectiveness and strong generalization ability of VLS. The code is available at https://github.com/XiaoqianRuan1/VLS.

1 Introduction

Reconstructing the 3D shape of an object from a single image is a fundamental problem in computer vision. It is inherently ill-posed, but recent advances in large-scale datasets [1, 2] and deep neural architectures [3–6] have led to models that can reconstruct the shapes of a wide range of common object categories from a single view. Despite this progress, existing works primarily focus on recovering whole-object shapes while ignoring their semantic parts.

Physical objects are typically composed of various semantic parts (e.g., Figure 1), together forming a hierarchical structure known as *partonomy* [7]. Understanding this part-whole structure is essential for fine-grained applications of 3D reconstruction across diverse domains. In robotics, perceiving object partonomy allows robots to manipulate or disassemble objects by recognizing their functional parts and spatial relationships [8–10]. In augmented reality, partonomic reconstruction enables systems to highlight or animate individual components of objects in the physical environment for interactive visualization and guided workflows [11–13].

Existing point cloud segmentation methods [14–19] can decompose an object's point cloud into semantic parts. Although one could feed the reconstructed shape from a single-view reconstruction model into a point cloud segmentation network, this pipeline faces several challenges. First, the

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

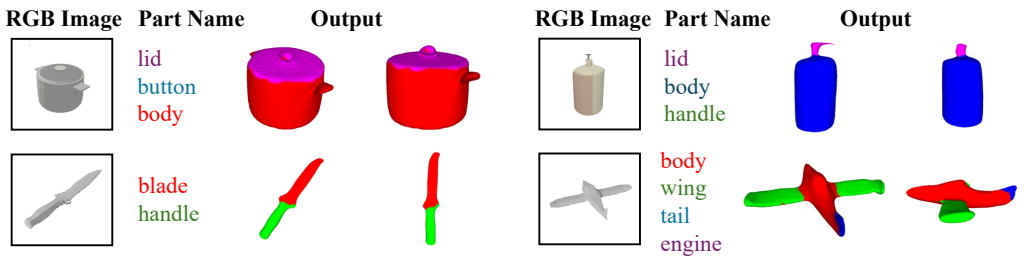

Figure 1: **Open-Vocabulary Partonomic 3D Reconstruction.** Given a single image and a set of part names, the goal is to reconstruct both the shape of an object and its constituent semantic parts, even when the object and part categories are unseen during training.

discrepancy between the ground truth point clouds used during training and the reconstructed shapes introduces a domain gap that may degrade 3D segmentation accuracy. Second, segmenting parts solely in 3D space ignores visual cues, such as color and texture, that are often crucial for understanding object partonomy and differentiating between geometrically similar but functionally distinct parts. Third, training and deploying two separate models increases computational cost and prevents mutual learning: neither model benefits from the other's intermediate representations. Recently, Ruan et al. [20] propose a compositional model to learn partonomic reconstruction from an image collection, but it is trained per category in a closed vocabulary and limited to a fixed mesh topology.

This paper introduces open-vocabulary partonomic 3D reconstruction, a novel problem setting illustrated in Figure 1. Given a single image and a set of part names, the goal is to reconstruct both the shape of an object and its constituent semantic parts, even when the object and part categories are unseen in training.

We propose a vision-language-shape (VLS) model to address this task. VLS jointly models the implicit representation of an object shape and the textual embedding of parts within an image-conditioned neural coordinate field. Thus, it can capture shapes with arbitrary topology and encode parts of any semantic category and spatial configuration. VLS further grounds each 3D coordinate to its corresponding 2D pixel context through a deformable implicit function. This allows it to accurately reconstruct both the overall object shape and its constituent parts from a single image.

To train the VLS with limited 3D part-labeled data and enable it with open-vocabulary capability, we propose an omni-supervised learning framework that leverages heterogeneous datasets with different levels of annotation and open-world knowledge from existing vision-language models. It is a two-stage training paradigm. In the first stage, VLS is trained on large-scale shape data augmented with 2D-to-3D lifted vision-language embeddings, establishing a shared latent space of textual concepts, image features, and geometry. In the second stage, the model is fine-tuned on a small 3D part-labeled dataset to achieve open-vocabulary partonomic 3D reconstruction.

It is worth noting that, unlike shape abstraction [21–25] or part-aware 3D generation [26–29], we focus on *semantic* part reconstruction in an open-vocabulary setting. Shape abstraction decomposes 3D objects into geometric primitives, such as cuboids, superquadrics, or convexes, while part-aware 3D generative models generate shapes together with class-agnostic parts. Neither can specify, identify, or control the parts' semantic categories, such as chair seat or mug handle. This distinction is crucial for applications that require both geometric reconstruction and fine-grained semantic perception. For example, a robot assisting a person in sitting down must identify the chair seat as the sitting surface and the chair back as the part that supports the body.

The contributions of this paper are summarized below.

- To our knowledge, this is the first work on open-vocabulary partonomic 3D reconstruction from a single image. It will facilitate applications requiring fine-grained 3D perception, e.g., robot manipulation and augmented reality.
- We propose a vision-language-shape (VLS) model that unifies vision, language, and shape representations within a shared neural field and grounds continuous 3D coordinates to 2D pixel context via a deformable implicit function. It can effectively reconstruct any-topology shapes and open-vocabulary parts from a single image.

- We propose an omni-supervised learning framework to train VLS with limited 3D part-labeled data by leveraging heterogeneous datasets with different levels of annotation and open-world knowledge from vision-language models.
- Extensive experiments on ShapeNetPart [30], PartNet [14, 31], and Objaverse [1] demonstrate the effectiveness and generalizability of VLS.

2 Related Work

Single-View 3D Reconstruction. Single-view 3D reconstruction methods are mainly generative-based [32–38] or regression-based [3, 39–42]. Generative methods use diffusion or generative priors to synthesize multi-view images or 3D representations. Zero123 [32] generates novel-view images from a single image, while One-2-3-45 [43] and One-2-3-45++ [44] generate textured meshes from synthetic views using SparseNeuS [45] and 2D/3D diffusion models. Magic123 [46] combines NeRF-based volume learning [47] with differentiable mesh optimization. In contrast, regression-based methods directly map 2D features to 3D structures. LRM [39] uses a DINO-based encoder [48] and an image-to-triplane decoder, while ZeroShape [3] predicts visible geometry and completes the full shape. TriPoSR [41] and Sparp [42] improve triplane encoding and sparse representation learning for efficient reconstruction.

Point Cloud Part Segmentation. Early point cloud part segmentation methods assume closed-set categories [49–51], with PointNet [50] and successors [51–53] predicting per-point labels directly. Recent open-vocabulary methods [19, 54–59] use text prompts to segment semantic parts. Many approaches distill 2D VLM outputs, such as GLIP [60] and SAM [61], into 3D space [17, 62, 63]. PartSLIP [14] fuses multi-view GLIP detections via 3D voting and grouping, while PartSLIP++ [64] uses SAM masks and EM-based lifting. PartDistill [17] trains a student model from VLM-generated 2D predictions. Other methods use CLIP embeddings directly: PointCLIP [16] aligns multi-view depth features with text, PointCLIPV2 [15] extends this to part segmentation with GPT-assisted prompts [65], and Find3D [18] improves performance through a new data engine.

Vision-Language Models and Open-Vocabulary Recognition. Vision-language models align visual and textual modalities in a shared embedding space [66–71]. CLIP [72] and ALIGN [73] use contrastive learning, including symmetric InfoNCE loss [74], to learn image-text alignment. Their strong generalization has enabled open-vocabulary detection [60, 75], semantic segmentation [76–79], and part segmentation [80–82]. Grounding DINO [75] further unifies detection and grounding through phrase-level supervision, language-guided query selection, and cross-modal decoding.

3 Method

This paper addresses open-vocabulary partonomic 3D reconstruction. Given a single image and a set of part names, the goal is to reconstruct both the shape of an object and its constituent semantic parts, even when the object and part categories are unseen during training. The input follows the open-vocabulary semantic or part segmentation setting [76–82], while the output extends the prediction from 2D to 3D. The part names can be queried by downstream applications (e.g., a robot cleaning the top of a table) or specified based on the recognized object category (e.g., a knife is typically composed of a handle and a blade). Our approach consists of a vision-language-shape model and an omni-supervised learning framework, which are described in Section 3.1 and Section 3.2, respectively.

3.1 Vision-Language-Shape (VLS) Model

VLS models vision, language, and shape representations within a unified neural field and grounds continuous 3D coordinates to 2D pixel context via a deformable implicit function. It enables accurate reconstruction of any-topology shapes and open-vocabulary parts from a single image. Figure 2 provides an overview of VLS.

Input Encoding. Given a single RGB image and a set of part names (e.g., "leg of chair" and "back of chair"), we use an image encoder and a text encoder to obtain visual embeddings $\boldsymbol{E}^{\text{vis}} \in \mathbb{R}^{C \times H \times W}$ and text embeddings $\boldsymbol{E}^{\text{lan}} \in \mathbb{R}^{C \times K}$, respectively. Here, C is the embedding dimension, and H and

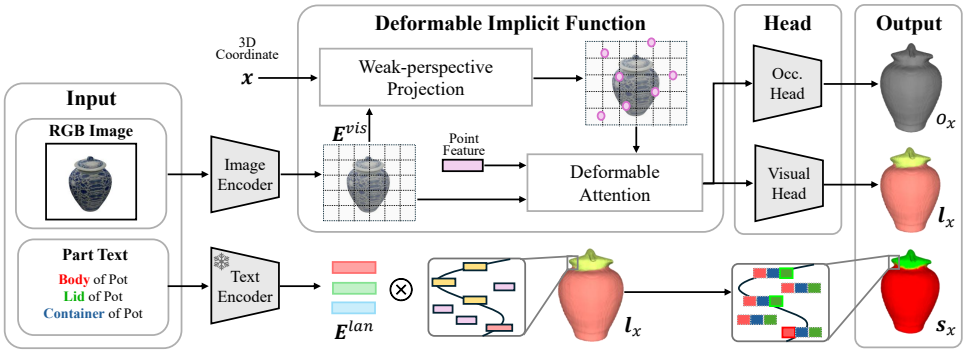

Figure 2: **Vision-Language-Shape (VLS) Model.** VLS jointly models the implicit representation of an object shape and the textual embedding of semantic parts within a unified neural field. It performs open-vocabulary partonomic 3D reconstruction by grounding continuous 3D coordinates to visual embeddings of an image and textual embeddings of part names.

W are the height and width of the visual embeddings. K is the number of part names, which is variable. Both image and text encoders are transformers [83] (detailed in Section 4.1).

Unified Neural Field. VLS jointly models the implicit representation of an object shape and the textual embedding of semantic parts within an image-conditioned neural coordinate field:

$$(o_{\boldsymbol{x}}, \boldsymbol{l}_{\boldsymbol{x}}) = f_\theta(\boldsymbol{x}; \boldsymbol{E}^{\text{vis}}),\ \boldsymbol{x} \in [0,1]^3 \tag{1}$$

$$\boldsymbol{s}_{\boldsymbol{x}} = \boldsymbol{l}_{\boldsymbol{x}}^T \boldsymbol{E}^{\text{lan}},\ \boldsymbol{x} \in [0,1]^3 \tag{2}$$

In Eq. (1), $o_{\boldsymbol{x}} \in [0,1]$ is the object surface occupancy at 3D point $\boldsymbol{x}$, $\boldsymbol{l}_{\boldsymbol{x}} \in \mathbb{R}^C$ is the semantic embedding at $\boldsymbol{x}$, and f_θ is a neural implicit function parametrized by θ. In Eq. (2), $\boldsymbol{s}_{\boldsymbol{x}} \in \mathbb{R}^K$ is the scores of the K part names at $\boldsymbol{x}$.

Conditioned on the visual embeddings $\boldsymbol{E}^{\text{vis}}$, f_θ maps an arbitrary 3D point coordinate $\boldsymbol{x}$ to its occupancy value and semantic embedding. A straightforward choice of f_θ is a multi-layer perceptron, but we will explore a more effective alternative below. The continuous field $\{(o_{\boldsymbol{x}}, \boldsymbol{s}_{\boldsymbol{x}}) : \boldsymbol{x} \in [0,1]^3\}$ characterizes both the shape of an object and the semantics and spatial distribution of each part.

Deformable Implicit Function. As illustrated in Figure 2, we instantiate f_θ as a deformable implicit function (DIFu):

$$f_\theta(\boldsymbol{x}; \boldsymbol{E}^{\text{vis}}) = \texttt{DeformAtt}(\boldsymbol{q}_{\boldsymbol{x}}, \boldsymbol{K}\boldsymbol{x}, \boldsymbol{E}^{\text{vis}}),\ \boldsymbol{x} \in [0,1]^3 \tag{3}$$

where `DeformAtt` denotes deformable attention [84]. $\boldsymbol{q}_{\boldsymbol{x}}$ is the features of a 3D point $\boldsymbol{x}$. It is initialized as the position embeddings and updated by deformable attention. $\boldsymbol{K}$ is a weak-perspective projection matrix. The projection of the 3D point coordinate to the 2D pixel location $\boldsymbol{K}\boldsymbol{x}$ serves as the reference point on the visual feature map. The detailed formula of deformable attention is in the appendix.

Discussion. By modeling the object surface and part textual embeddings within a unified neural implicit field, VLS can represent shapes with arbitrary topology and encode parts of any semantic categories and spatial configurations. This expressiveness in both geometry and semantics is essential for open-vocabulary partonomic reconstruction. The continuous coordinate field $\{(o_{\boldsymbol{x}}, \boldsymbol{s}_{\boldsymbol{x}}) : \boldsymbol{x} \in [0,1]^3\}$ can be converted into a mesh with per-vertex part labels using standard marching cubes.

The deformable implicit function (DIFu) integrates the strengths of deformable attention [84] and pixel-aligned implicit functions [85]. On one hand, DIFu anchors each 3D query point to its projected 2D pixel location, enabling VLS to retrieve visual features directly aligned with each query point. On the other hand, DIFu allows adaptive deformation of projected pixel locations, enabling VLS to leverage richer context of pixel embeddings while improving robustness against the inaccuracies of weak-perspective projection.

3.2 Omni-Supervised Learning

Existing 3D part-labeled datasets, e.g., ShapeNetPart [30] and PartNet [14, 31], are limited in both size and category diversity. End-to-end supervised learning on these datasets cannot generalize well to a wide range of object shapes and part categories, particularly those unseen during training.

To address this challenge, we introduce an omni-supervised learning framework that leverages heterogeneous datasets with different levels of annotation and open-world knowledge from existing vision-language models to train VLS. It consists of two stages: vision-language-shape pre-training and partonomic reconstruction fine-tuning, which are respectively described below.

Vision-Language-Shape Pre-Training. We pre-train VLS on large-scale shape datasets (ShapeNet [2] and Objaverse [1]) and 2D-to-3D lifted CLIP [72] features to learn a shared embedding space of textual concepts, image features, and object geometry.

As shown in Figure 3, we lift CLIP visual features from 2D to 3D through the following steps: (1) render a textured mesh into RGB images from multiple viewpoints; (2) feed each rendered image into CLIP's visual encoder to obtain a dense feature map; and (3) back-project each pixel's feature onto the mesh surface via ray casting, then average the projected features at each vertex.

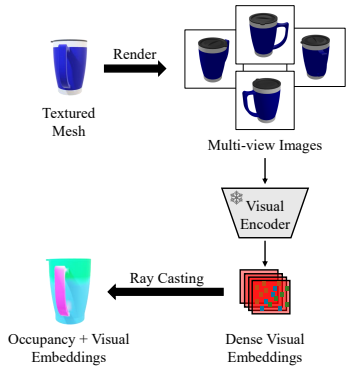

Figure 3: **Lift 2D CLIP Visual Embeddings to 3D.**

For each image, we have its ground truth mesh and the lifted CLIP features on the mesh, both centered within a unit cube in the camera view. We then sample a set of 3D points within this unit cube, denoted as $\mathcal{X} \subset [0,1]^3$. For each sampled point $\boldsymbol{x} \in \mathcal{X}$, we determine its occupancy $\hat{o}_{\boldsymbol{x}} \in \{0,1\}$ based on whether it lies on the object surface, and assign its CLIP features $\hat{\boldsymbol{l}}_{\boldsymbol{x}}$ according to the closest point on the surface. Our pre-training objective is formulated as:

$$\mathcal{L}^{\text{PT}} = \sum_{\boldsymbol{x} \in \mathcal{X}} (-\hat{o}_{\boldsymbol{x}} \log o_{\boldsymbol{x}} + \|\hat{\boldsymbol{l}}_{\boldsymbol{x}} - \boldsymbol{l}_{\boldsymbol{x}}\|_2^2) \tag{4}$$

where the first term is a binary cross-entropy loss and the second term is a mean squared error loss. We take the CLIP text encoder as our VLS text encoder and keep it frozen.

Partonomic Reconstruction Fine-Tuning. We adapt the VLS on small 3D part-labeled data (ShapeNetPart [30] and PartNet [14, 31]) to learn partonomic reconstruction. For an image in these datasets, we have its ground truth mesh and per-vertex part label. We use the text encoder to convert each part label into a textual embedding.

For each sampled point $\boldsymbol{x} \in \mathcal{X}$, we assign the part label of its closest surface point and then use the text encoder to obtain its textual embedding $\boldsymbol{l}_{\boldsymbol{x}}^+$; meanwhile, $\boldsymbol{l}_{\boldsymbol{x},k}^-$ denotes the textual embedding of an irrelevant part label. Our fine-tuning objective is formulated as a contrastive loss:

$$\mathcal{L}^{\text{FT}} = \sum_{\boldsymbol{x} \in \mathcal{X}} -\log \frac{\exp(\boldsymbol{l}_{\boldsymbol{x}}^T \boldsymbol{l}_{\boldsymbol{x}}^+)}{\exp(\boldsymbol{l}_{\boldsymbol{x}}^T \boldsymbol{l}_{\boldsymbol{x}}^+) + \sum_k \exp(\boldsymbol{l}_{\boldsymbol{x}}^T \boldsymbol{l}_{\boldsymbol{x},k}^-)} \tag{5}$$

It pushes $\boldsymbol{l}_{\boldsymbol{x}}$ toward the true semantic embedding while pushing it away from the irrelevant ones. We only fine-tune the head that predict $\boldsymbol{l}_{\boldsymbol{x}}$ while leaving the other parts of VLS frozen.

Discussion. The two-stage training framework shares the spirit of training recipes of modern LLMs [65, 69, 70] and VLMs [60, 72, 75, 86]: the first stage aims to train a foundational model on large-scale unlabeled or weakly labeled data for open-world, generic knowledge, while the second stage supervised fine-tune the foundational model on small-scale labeled data for specific tasks.

In the first stage, VLS learns an aligned vision-language-shape representation that captures generic part structures. In the second stage, the head serves as a lightweight adaptor that improves part discriminability of this aligned representation using limited 3D part labels, while preserving open-vocabulary generalization. Altogether, our omni-supervised learning framework takes the best of heterogeneous datasets with different levels of annotation and open-world knowledge from vision-language models.

4 Experiments

4.1 Setting

Data. Our VLS is pre-trained on ShapeNet [2] and Objaverse [1] (as well as using the CLIP model), and fine-tuned on ShapeNetPart [30] and PartNet [14, 31]. ShapeNet [2] is a large-scale 3D model repository spanning 55 object categories, with thousands of CAD models per category, each annotated with meshes. Objaverse [1] extends this scale dramatically, containing over 800K high-quality 3D assets spanning 1,156 categories collected from diverse sources. ShapeNetPart [30] is a curated subset of ShapeNet [2] focused on part-level segmentation, covering 16 object categories. It includes roughly 14K instances, each represented as a 2,048-point point cloud with part-level ground truth. PartNet [31] is a large-scale benchmark for fine-grained 3D object part segmentation with hierarchical part annotations. Liu et al. [14] extend PartNet by introducing a richer taxonomy of parts across 45 object categories but it contains only about 1,900 annotated instances per category. We use only the coarsest-level part annotations. To evaluate the generalizability of our model to unseen categories, we exclude 5 ShapeNetPart [30] categories and 11 PartNet [14, 31] categories from partonomic reconstruction fine-tuning. Thus, the fine-tuning data includes 11 object categories from ShapeNetPart [30] and 34 object categories from PartNet [14, 31], with averages of 3.2 and 3.5 part categories across object categories, respectively. Our testing data contains 16 object categories from ShapeNetPart [30], including 5 unseen categories, and 45 categories from PartNet [14, 31], including 11 unseen categories. The average numbers of part categories per object category are 3.1 for ShapeNetPart [30] and 3.3 for PartNet [14, 31], for unseen categories, the average numbers of part categories are 2.8 for both datasets.

Evaluation Metrics. We evaluate shape reconstruction and semantic part decomposition using Chamfer Distance (CD) [87] and mean Intersection-over-Union (mIoU) [88], respectively. CD [87] calculates the symmetric mean squared distance between points on the predicted and ground truth surfaces. mIoU [88] averages the IoU scores over all semantic part labels within each object category and then across all categories.

Implementation Details. The image backbone and occupancy predictor of VLS follow ZeroShape [3], a state-of-the-art single-view shape regression network, due to its simple design, strong performance, and training/inference efficiency. The text encoder follows the CLIP text encoder [72]. VLS is implemented in PyTorch [89] and optimized via the Adam optimizer [90]. It is pre-trained for 15 epochs and fine-tuned for 50 epochs. In pre-training, the weights for shape and semantic losses are 1 and 0.1, respectively. During fine-tuning, the backbone and the occupancy prediction branch are frozen. The learning rate is set to 3×10^{-5} for both stages. Each shape is rendered into 24 RGB images from uniformly sampled camera poses surrounding the object. The input image resolution is set to 224×224. During training, we sample 4096 points from each shape. During inference, we convert the implicit shape and part representation into a segmented mesh using marching cubes with 32^3 voxels. Unless otherwise specified, we use part names defined for each object category in the datasets. We also validate VLS under different part input settings, including noisy part names, ChatGPT-generated names, and long descriptive prompts (available in appendix). We train our model on $2\times$ NVIDIA L40S GPUs, which takes about 2 days for pre-training and about 1 day for fine-tuning.

4.2 Baselines

Since no prior work has addressed the same task, we construct baselines by integrating existing single-view shape reconstruction models with 3D part segmentation methods: the shape is first predicted from the input image and then used for 3D part segmentation. We also report performance using ground truth shape and shape reconstructed by our model. Specifically, we use ZeroShape [3] for shape prediction and apply five different 3D part segmentation methods, including four point-cloud part segmentation methods: PointCLIPv2 [15], PartSLIP++ [64], ZeroPS [19], and Find3D [18], and one mesh part segmentation methods: SATR [63]. These 3D part segmentation methods typically use SAM [61] and GLIP [60] to obtain zero-shot 2D segmentations, which are then lifted to 3D masks. Since these 2D foundation models, particularly SAM, were trained on nearly all common object categories with fine-grained part mask annotations [61], the 3D part segmentation methods

Table 1: **Quantitative Results on the ShapeNetPart Dataset.**

Method	Unseen Categories mIoU ↑	Unseen Categories CD ↓	Seen Categories mIoU ↑	Seen Categories CD ↓
ZeroShape + PointCLIPV2	20.75	0.277	20.36	0.325
Our Shape + PointCLIPV2	20.63	0.274	20.16	0.319
GT Shape + PointCLIPV2	20.65	0	20.07	0
ZeroShape + PartSLIP++	20.13	0.277	19.69	0.325
Our Shape + PartSLIP++	20.23	0.274	19.80	0.319
GT Shape + PartSLIP++	20.74	0	19.98	0
ZeroShape + SATR	23.78	0.277	21.31	0.325
Our Shape + SATR	23.89	0.274	21.44	0.319
GT Shape + SATR	26.61	0	25.05	0
ZeroShape + ZeroPS	20.15	0.277	19.67	0.325
Our Shape + ZeroPS	20.27	0.274	19.78	0.319
GT Shape + ZeroPS	20.76	0	19.97	0
ZeroShape + Find3D	25.56	0.277	23.76	0.325
Our Shape + Find3D	26.01	0.274	23.83	0.319
GT Shape + Find3D	28.77	0	28.00	0
Ours	**28.87**	**0.274**	**30.84**	**0.319**

Table 2: **Quantitative Results on the PartNet Dataset.**

Method	Unseen Categories mIoU ↑	Unseen Categories CD ↓	Seen Categories mIoU ↑	Seen Categories CD ↓
ZeroShape + PointCLIPV2	15.63	0.210	15.03	0.289
Our Shape + PointCLIPV2	16.14	0.171	15.92	0.200
GT Shape + PointCLIPV2	17.78	0	17.09	0
ZeroShape + PartSLIP++	19.57	0.210	18.56	0.289
Our Shape + PartSLIP++	19.02	0.171	18.69	0.200
GT Shape + PartSLIP++	19.33	0	18.61	0
ZeroShape + SATR	18.42	0.210	17.58	0.289
Our Shape + SATR	18.54	0.171	17.73	0.200
GT Shape + SATR	20.50	0	19.12	0
ZeroShape + ZeroPS	19.74	0.210	18.63	0.289
Our Shape + ZeroPS	19.84	0.171	18.44	0.200
GT Shape + ZeroPS	19.83	0	18.29	0
ZeroShape + Find3D	26.72	0.210	23.48	0.289
Our Shape + Find3D	27.04	0.171	23.99	0.200
GT Shape + Find3D	28.00	0	25.67	0
Ours	**35.89**	**0.171**	**39.41**	**0.200**

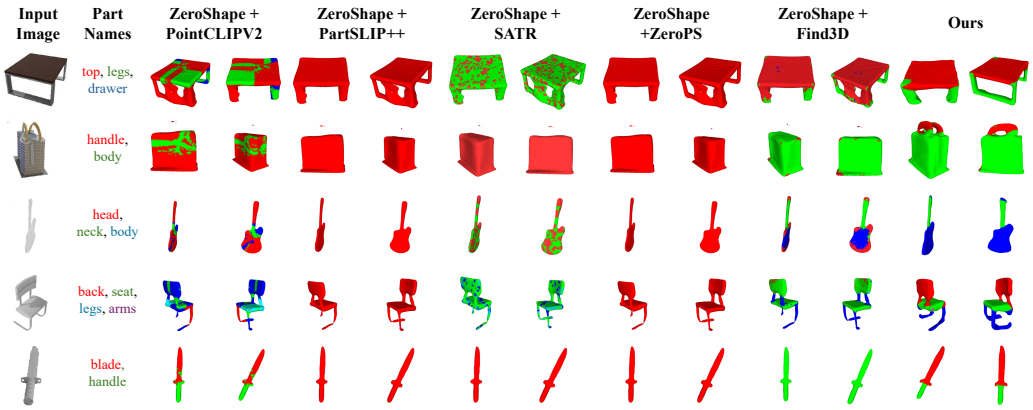

Figure 4: **Qualitative Results on the ShapeNetPart Dataset.** VLS is compared with ZeroShape + 3D part segmentation methods. The top three rows are unseen categories while the bottom two rows are seen categories.

often demonstrate strong generalizability to unseen categories in our setting, making them suitable baselines for validating VLS.

4.3 Main Results

Results on ShapeNetPart. Table 1 shows the comparison between VLS and other methods under both seen and unseen categories. VLS achieves the best performance across all metrics, significantly outperforming the combination of existing 3D shape reconstruction and 3D part segmentation methods. Particularly, VLS improves mIoU by a clear margin on both seen and unseen categories, while maintaining competitive CD for shape reconstruction. In addition, replacing ZeroShape [3] with our reconstructed shapes consistently improves segmentation performance across different methods, demonstrating the advantage of our reconstructed shape.

Figure 4 presents qualitative results on the ShapeNetPart dataset. It demonstrates that VLS consistently generates clean, coherent, and semantically accurate part masks, while the baselines either collapse into a single part or fail to distinguish meaningful part boundaries. For example, VLS distinctly identifies the handle and body, showing the robustness to thin structures and unseen parts layouts. However, PointCLIPV2 [15] and Find3D [18] tend to produce cluttered and inconsistent masks. PartSLIP++ [64], SATR [63] and ZeroPS [19] are unable to segment the semantic parts and treat the whole object as a part.

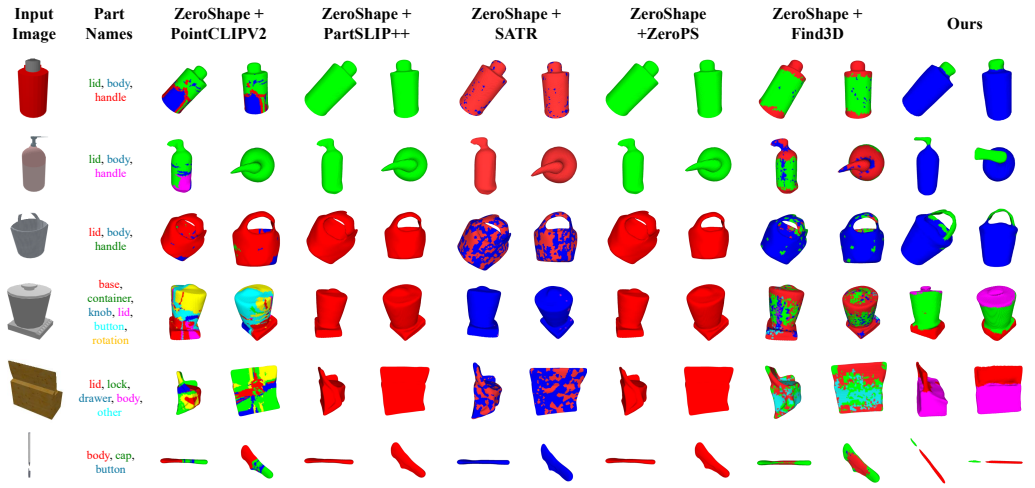

Figure 5: **Qualitative Results on the PartNet Dataset.** VLS is compared with ZeroShape + 3D part segmentation methods. The top three rows are unseen categories while the bottom three rows are seen categories.

Results on PartNet. Table 2 summarizes quantitative results on PartNet [14, 31]. VLS outperforms all other methods on both shape reconstruction and part decomposition metrics (CD and mIoU), with particularly strong gains on unseen categories. Compared to methods built on Zeroshape [3], our proposed method consistently provides better geometric representations which leads to better part segmentation results. These results show the effectiveness of our unified vision-language-shape representation for generalizable partonomic reconstruction.

Figure 5 presents qualitative results on both seen and unseen categories. Given the input image and part-name queries, VLS is capable to produce more coherent and semantically aligned part predictions compared with the combination of ZeroShape [3] and PointCLIPV2 [15], PartSLIP++ [64], SATR [63], ZeroPS [19] and Find3D [18]. PointCLIPV2 [15] tends to generate noisy part masks, particularly for thin structures such as handles or lid. PartSLIP++ [64], SATR [63] and ZeroPS [19] segment parts based on 2D semantic predictions and both these methods could not well separate the semantic parts and would consider the whole object as a part. Find3D [18] partially improves geometric alignment but still suffers from semantic ambiguity, tending to mix multiple parts within a single region. In contrast, VLS accurately segments these parts while maintaining geometric completeness and cross-part consistency. In addition, our model is able to reconstruct more accurate shapes compared with ZeroShape [3], like sharper boundaries.

Results on Objaverse. Figure 6 shows qualitative comparisons between VLS and the combination of ZeroShape [3] with two 3D part segmentation methods, including PointCLIPV2 [15] and Find3D [18] on Objaverse. VLS demonstrates superior shape reconstruction and produces part decompositions with clearer boundaries and more semantically meaningful segments for each input image and its corresponding part names. In contrast, PointCLIPV2 [15] yields noisy and fragmented predictions, while Find3D [18] suffers from semantic ambiguity and inconsistent labeling across regions. Quantitative results are in the appendix.

Inference Time. Table 3 compares the inference time of VLS and other methods, measured on an NVIDIA L40S GPU. VLS achieves the fastest runtime at 4.2 seconds. This efficiency stems from performing partonomic reconstruction directly from an image, rather than first reconstructing a single-view shape and then segmenting the resulting point cloud.

Table 3: **Inference Time.**

Method	Runtime (s) ↓
ZeroShape + PointCLIPV2	4.6
ZeroShape + PartSLIP++	85.0
ZeroShape + ZeroPS	68.9
ZeroShape + SATR	59.6
ZeroShape + Find3D	4.5
Ours	**4.2**

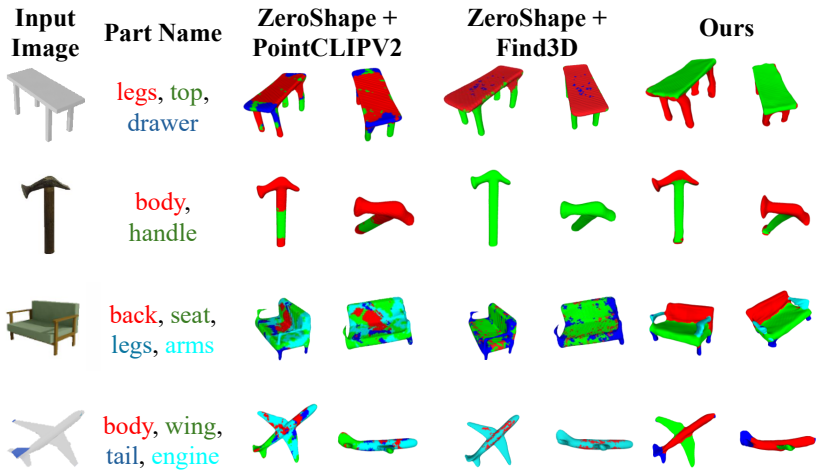

Figure 6: **Qualitative Results on the Objaverse Dataset.** VLS is compared with ZeroShape + PointCLIPV2. The top two rows are unseen categories while the bottom two rows are seen categories.

4.4 Ablation Study

We conduct ablation experiments on the PartNet dataset.

Impact of Each Component. Table 4 reports ablation results validating the necessity of DIFu, pre-training (PT), and fine-tuning (FT). Without DIFu means replacing it with pixel-aligned 3D-to-2D projections without deformable attention. Without pre-training means directly training the model on the 3D part-labeled data. Without FT means using the model simply trained on 3D CLIP embeddings. Table 4 indicates that both pre-training and fine-tuning contribute significantly to the strong performance of VLS, which is further consolidated by DIFu.

Table 4: **Ablation Study on Each Component.**

DIFu	PT	FT	Unseen		Seen	
			mIoU ↑	CD ↓	mIoU ↑	CD ↓
	✓	✓	29.12	0.192	31.33	0.225
✓	✓		31.06	**0.171**	27.02	**0.200**
✓		✓	23.41	0.187	28.73	0.215
✓	✓	✓	**35.89**	**0.171**	**39.41**	**0.200**

Fine-Tuning Strategies. Table 5 compares the performance of fine-tuning the full model or only the part head during the second stage of our omni-supervised learning framework. The results reveal a clear trade-off between seen and unseen category performance. While full fine-tuning improves performance on seen categories, it substantially degrades generalization to unseen ones. Since open-vocabulary settings prioritize generalizability to unseen categories, we adopt part head fine-tuning.

Table 5: **Fine-Tuning Strategies.**

	Unseen		Seen	
	mIoU ↑	CD ↓	mIoU ↑	CD ↓
Full FT	33.73	0.180	**41.36**	**0.195**
Head FT	35.89	**0.171**	39.41	0.200

Impact of Language Supervision on Shape. We remove the 3D CLIP loss, i.e., the second term in Eq. (4), during pre-training and find that this degrades the shape CD, as shown in Table 6. This suggests that language supervision anchors shape reconstruction to semantically meaningful structures, leading to more geometrically accurate outputs.

Table 6: **Impact of 3D CLIP Loss on CD.**

	Unseen	Seen
w/o 3D CLIP Loss	0.187	0.215
Ours	**0.171**	**0.200**

Noisy Part Names. Table 7 evaluates the robustness of VLS to noisy part names on unseen categories. We construct noisy part queries by randomly inserting three irrelevant part names. Despite this perturbation, VLS remains effective, achieving 32.28% mIoU compared to 35.89% mIoU with clean part names, while maintaining shape quality. The degradation is expected, as irrelevant part names introduce spurious segments that reduce overall mIoU. Nevertheless, the results demonstrate that VLS is generally robust to noisy part queries.

Table 7: **Noisy Part Names.**

	mIoU	CD
Clean Part Names	**35.89**	**0.171**
Noisy Part Names	32.28	**0.171**

Visualization of Lifted 3D CLIP Embeddings. Figure 7 visualizes the lifted 3D CLIP features of unseen categories using PCA, providing insights into the semantic representations learned by our framework. For most categories, such as pot, basket, and table, the learned features exhibit clear spatial and semantic structures, with different object parts showing distinguishable feature distributions. However, for certain categories, such as switch, the feature distributions of different parts are less distinguishable, indicating limited semantic discrimination between individual components. Such ambiguity makes it challenging to accurately identify and separate different parts, resulting in degraded partonomic reconstruction performance.

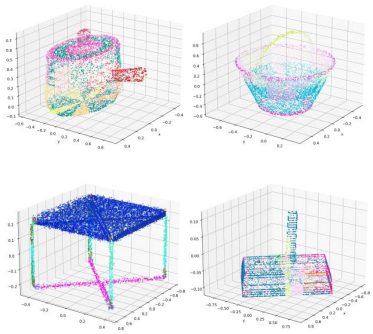

Figure 7: **PCA Visualizations of Lifted 3D CLIP Embeddings.**

4.5 Real-World Images

To further evaluate the generalization capability of VLS beyond synthesis images, we conduct qualitative experiments on real-world images, as illustrated in Figure 8. Given a single RGB image and a set of part names, VLS jointly reconstructs the overall object geometry and identifies its constituent semantic parts. As for the hammer, VLS accurately separates the head from the handle, demonstrating its ability to recognize functionally distinct components. These visualization examples demonstrate the potential of VLS to generalize from synthetic training data to real-world images and reconstructs geometrically coherent objects with meaningful semantic parts across diverse categories.

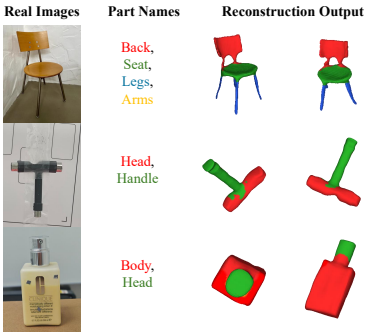

Figure 8: **Qualitative Results on Real-World Images.**

5 Conclusion

This paper proposes a vision-language-shape model (VLS) for open-vocabulary partonomic reconstruction and introduces an omni-supervised learning framework to train VLS on limited 3D part-labeled data. Experimental results show that VLS can effectively reconstruct both object shapes and semantic parts from a single image, even for unseen categories. It also demonstrates better performance than methods that integrate single-view shape reconstruction with 3D part segmentation.

Broader Impacts. This work has the potential to facilitate applications requiring fine-grained 3D object perception, such as robotic manipulation and augmented reality. However, these applications may introduce potential risks. In augmented reality, capturing images of users' surroundings could inadvertently expose private spaces. These privacy risks primarily arise from the sensing and deployment pipeline rather than the reconstruction model itself, and could be mitigated through strict access controls and on-device processing. In robotics, inaccurate part reconstruction could lead to unintended manipulation decisions, such as grasping the wrong component. Potential mitigation strategies include confidence-aware planning and conservative failure handling.

Acknowledgments and Disclosure of Funding

This work was supported in part by National Science Foundation (NSF) grants IIS-2442540 and ECCS-2400900. This work used the Delta system at the National Center for Supercomputing Applications (NSF grant OAC-2005572) through the Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support (ACCESS) program, which is supported by NSF grants #2138259, #2138286, #2138307, #2137603, and #2138296.

References

[1] 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.

[2] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[3] Zixuan Huang, Stefan Stojanov, Anh Thai, Varun Jampani, and James M Rehg. Zeroshape: Regression-based zero-shot shape reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10061–10071, 2024.

[4] Zixuan Huang, Mark Boss, Aaryaman Vasishta, James M Rehg, and Varun Jampani. Spar3d: Stable point-aware reconstruction of 3d objects from single images. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 16860–16870, 2025.

[5] Hanwen Jiang, Qixing Huang, and Georgios Pavlakos. Real3d: Towards scaling large reconstruction models with real images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5821–5833, 2025.

[6] Andreea Ardelean, Mert Özer, and Bernhard Egger. Gen3dsr: Generalizable 3d scene reconstruction via divide and conquer from a single view. In *2025 International Conference on 3D Vision (3DV)*, pages 616–626. IEEE, 2025.

[7] Simone Pribbenow. Meronymic relationships: From classical mereology to complex part-whole relations. In *The semantics of relationships: An interdisciplinary perspective*, pages 35–50. Springer, 2002.

[8] Yuanchen Ju, Kaizhe Hu, Guowei Zhang, Gu Zhang, Mingrun Jiang, and Huazhe Xu. Robo-abc: Affordance generalization beyond categories via semantic correspondence for robot manipulation. In *European Conference on Computer Vision*, pages 222–239. Springer, 2024.

[9] Gen Li, Nikolaos Tsagkas, Jifei Song, Ruaridh Mon-Williams, Sethu Vijayakumar, Kun Shao, and Laura Sevilla-Lara. Learning precise affordances from egocentric videos for robotic manipulation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10581–10591, 2025.

[10] Weiyu Liu, Jiayuan Mao, Joy Hsu, Tucker Hermans, Animesh Garg, and Jiajun Wu. Composable part-based manipulation. *arXiv preprint arXiv:2405.05876*, 2024.

[11] Safoura Rezapour Lakani, Antonio J Rodríguez-Sánchez, and Justus Piater. Towards affordance detection for robot manipulation using affordance for parts and parts for affordance. *Autonomous Robots*, 43(5):1155–1172, 2019.

[12] Bogdan Moldovan, Plinio Moreno, Martijn Van Otterlo, José Santos-Victor, and Luc De Raedt. Learning relational affordance models for robots in multi-object manipulation tasks. In *2012 ieee international conference on robotics and automation*, pages 4373–4378. IEEE, 2012.

[13] Yiran Geng, Boshi An, Haoran Geng, Yuanpei Chen, Yaodong Yang, and Hao Dong. Rlafford: End-to-end affordance learning for robotic manipulation. In *2023 IEEE International conference on robotics and automation (ICRA)*, pages 5880–5886. IEEE, 2023.

[14] 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.

[15] 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.

[16] 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.

[17] 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.

[18] Ziqi Ma, Yisong Yue, and Georgia Gkioxari. Find any part in 3d. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7818–7827, 2025.

[19] Yuheng Xue, Nenglun Chen, Jun Liu, and Wenyun Sun. Zerops: High-quality cross-modal knowledge transfer for zero-shot 3d part segmentation. In *2025 International Conference on 3D Vision (3DV)*, pages 1328–1339. IEEE, 2025.

[20] Xiaoqian Ruan, Pei Yu, Dian Jia, Hyeonjeong Park, Peixi Xiong, and Wei Tang. Learning partonomic 3d reconstruction from image collections. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 26734–26744, 2025.

[21] Shubham Tulsiani, Hao Su, Leonidas J Guibas, Alexei A Efros, and Jitendra Malik. Learning shape abstractions by assembling volumetric primitives. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2635–2643, 2017.

[22] Despoina Paschalidou, Ali Osman Ulusoy, and Andreas Geiger. Superquadrics revisited: Learning 3d shape parsing beyond cuboids. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10344–10353, 2019.

[23] R Kenny Jones, Paul Guerrero, Niloy J Mitra, and Daniel Ritchie. Shapecoder: Discovering abstractions for visual programs from unstructured primitives. *ACM Transactions on Graphics (TOG)*, 42(4):1–17, 2023.

[24] Weixiao Liu, Yuwei Wu, Sipu Ruan, and Gregory S Chirikjian. Marching-primitives: Shape abstraction from signed distance function. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8771–8780, 2023.

[25] Sunkyung Park, Jeongmin Lee, and Dongjun Lee. Shape abstraction via marching differentiable support functions. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 16902–16911, 2025.

[26] Yuchen Lin, Chenguo Lin, Panwang Pan, Honglei Yan, Yiqiang Feng, Yadong Mu, and Katerina Fragkiadaki. Partcrafter: Structured 3d mesh generation via compositional latent diffusion transformers. *arXiv preprint arXiv:2506.05573*, 2025.

[27] Jiaxiang Tang, Ruijie Lu, Max Li, Zekun Hao, Xuan Li, Fangyin Wei, Shuran Song, Gang Zeng, Ming-Yu Liu, and Tsung-Yi Lin. Efficient part-level 3d object generation via dual volume packing. *Advances in Neural Information Processing Systems*, 38:27115–27137, 2026.

[28] Minghao Chen, Jianyuan Wang, Roman Shapovalov, Tom Monnier, Hyunyoung Jung, Dilin Wang, Rakesh Ranjan, Iro Laina, and Andrea Vedaldi. Autopartgen: Autogressive 3d part generation and discovery. *arXiv preprint arXiv:2507.13346*, 2025.

[29] Yunhan Yang, Yufan Zhou, Yuan-Chen Guo, Zi-Xin Zou, Yukun Huang, Ying-Tian Liu, Hao Xu, Ding Liang, Yan-Pei Cao, and Xihui Liu. Omnipart: Part-aware 3d generation with semantic decoupling and structural cohesion. In *Proceedings of the SIGGRAPH Asia 2025 Conference Papers*, pages 1–12, 2025.

[30] 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.

[31] 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.

[32] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3d object. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9298–9309, 2023.

[33] Stanislaw Szymanowicz, Christian Rupprecht, and Andrea Vedaldi. Viewset diffusion:(0-) image-conditioned 3d generative models from 2d data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8863–8873, 2023.

[34] Xin Kong, Shikun Liu, Xiaoyang Lyu, Marwan Taher, Xiaojuan Qi, and Andrew J Davison. Eschernet: A generative model for scalable view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9503–9513, 2024.

[35] Zixin Zou, Weihao Cheng, Yan-Pei Cao, Shi-Sheng Huang, Ying Shan, and Song-Hai Zhang. Sparse3d: Distilling multiview-consistent diffusion for object reconstruction from sparse views. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 7900–7908, 2024.

[36] Yufei Ye, Poorvi Hebbar, Abhinav Gupta, and Shubham Tulsiani. Diffusion-guided reconstruction of everyday hand-object interaction clips. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19717–19728, 2023.

[37] Rameen Abdal, Wang Yifan, Zifan Shi, Yinghao Xu, Ryan Po, Zhengfei Kuang, Qifeng Chen, Dit-Yan Yeung, and Gordon Wetzstein. Gaussian shell maps for efficient 3d human generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9441–9451, 2024.

[38] Jeong-gi Kwak, Erqun Dong, Yuhe Jin, Hanseok Ko, Shweta Mahajan, and Kwang Moo Yi. Vivid-1-to-3: Novel view synthesis with video diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6775–6785, 2024.

[39] Yicong Hong, Kai Zhang, Jiuxiang Gu, Sai Bi, Yang Zhou, Difan Liu, Feng Liu, Kalyan Sunkavalli, Trung Bui, and Hao Tan. Lrm: Large reconstruction model for single image to 3d. *arXiv preprint arXiv:2311.04400*, 2023.

[40] Zhengyi Wang, Yikai Wang, Yifei Chen, Chendong Xiang, Shuo Chen, Dajiang Yu, Chongxuan Li, Hang Su, and Jun Zhu. Crm: Single image to 3d textured mesh with convolutional reconstruction model. In *European Conference on Computer Vision*, pages 57–74. Springer, 2024.

[41] Dmitry Tochilkin, David Pankratz, Zexiang Liu, Zixuan Huang, Adam Letts, Yangguang Li, Ding Liang, Christian Laforte, Varun Jampani, and Yan-Pei Cao. Triposr: Fast 3d object reconstruction from a single image. *arXiv preprint arXiv:2403.02151*, 2024.

[42] Chao Xu, Ang Li, Linghao Chen, Yulin Liu, Ruoxi Shi, Hao Su, and Minghua Liu. Sparp: Fast 3d object reconstruction and pose estimation from sparse views. In *European Conference on Computer Vision*, pages 143–163. Springer, 2024.

[43] Minghua Liu, Chao Xu, Haian Jin, Linghao Chen, Mukund Varma T, Zexiang Xu, and Hao Su. One-2-3-45: Any single image to 3d mesh in 45 seconds without per-shape optimization. *Advances in Neural Information Processing Systems*, 36:22226–22246, 2023.

[44] Minghua Liu, Ruoxi Shi, Linghao Chen, Zhuoyang Zhang, Chao Xu, Xinyue Wei, Hansheng Chen, Chong Zeng, Jiayuan Gu, and Hao Su. One-2-3-45++: Fast single image to 3d objects with consistent multi-view generation and 3d diffusion. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10072–10083. IEEE, 2024.

[45] Xiaoxiao Long, Cheng Lin, Peng Wang, Taku Komura, and Wenping Wang. Sparseneus: Fast generalizable neural surface reconstruction from sparse views. In *European Conference on Computer Vision*, pages 210–227. Springer, 2022.

[46] Guocheng Qian, Jinjie Mai, Abdullah Hamdi, Jian Ren, Aliaksandr Siarohin, Bing Li, Hsin-Ying Lee, Ivan Skorokhodov, Peter Wonka, Sergey Tulyakov, et al. Magic123: One image to high-quality 3d object generation using both 2d and 3d diffusion priors. 2024.

[47] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

[48] Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel M Ni, and Heung-Yeung Shum. Dino: Detr with improved denoising anchor boxes for end-to-end object detection. *arXiv preprint arXiv:2203.03605*, 2022.

[49] Zongji Wang and Feng Lu. Voxsegnet: Volumetric cnns for semantic part segmentation of 3d shapes. *IEEE transactions on visualization and computer graphics*, 26(9):2919–2930, 2019.

[50] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[51] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017.

[52] 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.

[53] Xiaoyang Wu, Yixing Lao, Li Jiang, Xihui Liu, and Hengshuang Zhao. Point transformer v2: Grouped vector attention and partition-based pooling. *Advances in Neural Information Processing Systems*, 35:33330–33342, 2022.

[54] Xueyi Liu, Xiaomeng Xu, Anyi Rao, Chuang Gan, and Li Yi. Autogpart: Intermediate supervision search for generalizable 3d part segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11624–11634, 2022.

[55] Anh Thai, Weiyao Wang, Hao Tang, Stefan Stojanov, James M Rehg, and Matt Feiszli. $3\times$ 2: 3d object part segmentation by 2d semantic correspondences. In *European Conference on Computer Vision*, pages 149–166. Springer, 2024.

[56] 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. *arXiv preprint arXiv:2411.07184*, 2024.

[57] George Tang, William Zhao, Logan Ford, David Ben-Haim, and Paul Zhang. Segment any mesh: Zero-shot mesh part segmentation via lifting segment anything 2 to 3d. *CoRR*, 2024.

[58] Ayça Takmaz, Elisabetta Fedele, Robert W Sumner, Marc Pollefeys, Federico Tombari, and Francis Engelmann. Openmask3d: Open-vocabulary 3d instance segmentation. *arXiv preprint arXiv:2306.13631*, 2023.

[59] Minghua Liu, Mikaela Angelina Uy, Donglai Xiang, Hao Su, Sanja Fidler, Nicholas Sharp, and Jun Gao. Partfield: Learning 3d feature fields for part segmentation and beyond. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9704–9715, 2025.

[60] 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.

[61] 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.

[62] Hyunjin Kim and Minhyuk Sung. Partstad: 2d-to-3d part segmentation task adaptation. In *European Conference on Computer Vision*, pages 422–439. Springer, 2024.

[63] 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.

[64] 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.

[65] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[66] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pages 19730–19742. PMLR, 2023.

[67] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023.

[68] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407, 2024.

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

[70] Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024.

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

[72] 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.

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

[74] Evgenia Rusak, Patrik Reizinger, Attila Juhos, Oliver Bringmann, Roland S Zimmermann, and Wieland Brendel. Infonce: Identifying the gap between theory and practice. *arXiv preprint arXiv:2407.00143*, 2024.

[75] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In *European conference on computer vision*, pages 38–55. Springer, 2024.

[76] Jilan Xu, Junlin Hou, Yuejie Zhang, Rui Feng, Yi Wang, Yu Qiao, and Weidi Xie. Learning open-vocabulary semantic segmentation models from natural language supervision. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2935–2944, 2023.

[77] Feng Liang, Bichen Wu, Xiaoliang Dai, Kunpeng Li, Yinan Zhao, Hang Zhang, Peizhao Zhang, Peter Vajda, and Diana Marculescu. Open-vocabulary semantic segmentation with mask-adapted clip. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7061–7070, 2023.

[78] Mengde Xu, Zheng Zhang, Fangyun Wei, Han Hu, and Xiang Bai. Side adapter network for open-vocabulary semantic segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2945–2954, 2023.

[79] Golnaz Ghiasi, Xiuye Gu, Yin Cui, and Tsung-Yi Lin. Scaling open-vocabulary image segmentation with image-level labels. In *European conference on computer vision*, pages 540–557. Springer, 2022.

[80] Meng Wei, Xiaoyu Yue, Wenwei Zhang, Shu Kong, Xihui Liu, and Jiangmiao Pang. Ov-parts: Towards open-vocabulary part segmentation. *Advances in Neural Information Processing Systems*, 36:70094–70114, 2023.

[81] Peize Sun, Shoufa Chen, Chenchen Zhu, Fanyi Xiao, Ping Luo, Saining Xie, and Zhicheng Yan. Going denser with open-vocabulary part segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15453–15465, 2023.

[82] Jiho Choi, Seonho Lee, Seungho Lee, Minhyun Lee, and Hyunjung Shim. Understanding multi-granularity for open-vocabulary part segmentation. *Advances in Neural Information Processing Systems*, 37:137161–137189, 2024.

[83] Zhuofan Xia, Xuran Pan, Shiji Song, Li Erran Li, and Gao Huang. Vision transformer with deformable attention. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4794–4803, 2022.

[84] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020.

[85] Shunsuke Saito, Zeng Huang, Ryota Natsume, Shigeo Morishima, Angjoo Kanazawa, and Hao Li. Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2304–2314, 2019.

[86] Tianhe Ren, Qing Jiang, Shilong Liu, Zhaoyang Zeng, Wenlong Liu, Han Gao, Hongjie Huang, Zhengyu Ma, Xiaoke Jiang, Yihao Chen, et al. Grounding dino 1.5: Advance the" edge" of open-set object detection. *arXiv preprint arXiv:2405.10300*, 2024.

[87] Tong Wu, Liang Pan, Junzhe Zhang, Tai Wang, Ziwei Liu, and Dahua Lin. Balanced chamfer distance as a comprehensive metric for point cloud completion. *Advances in Neural Information Processing Systems*, 34:29088–29100, 2021.

[88] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 658–666, 2019.

[89] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[90] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[91] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020.

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

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

A Appendix

A.1 Deformable Attention

Given a visual feature map $\boldsymbol{E}$, and let $\boldsymbol{q}$ and $\boldsymbol{p}$ be respectively the features and reference position of a query, the deformable attention [84] is defined as

$$\texttt{DeformAtt}(\boldsymbol{q}, \boldsymbol{p}, \boldsymbol{E}) = \sum_{m=1}^{M} \boldsymbol{W}_m (\sum_{k=1}^{K} a_{mk} \boldsymbol{W}'_m \boldsymbol{E}(\boldsymbol{p}_m + \Delta \boldsymbol{p}_{mk})) \tag{6}$$

where M is the number of scales, K is the number of sampled deformations, and $\boldsymbol{W}_m$ and $\boldsymbol{W}'_m$ are learnable projection matrices. a_{mk} and $\Delta \boldsymbol{p}_{mk}$ are respectively an attention weight and a deformation offset, both predicted from the query features. $\boldsymbol{E}(\boldsymbol{p}_m + \Delta \boldsymbol{p}_{mk})$ is the sampled visual features using bilinear interpolation.

A.2 Quantitative Results on Objaverse

We evaluate VLS on Objaverse [1] using the subset at the intersection of part annotations from Find3D [18] and image–shape pairs from ZeroShape [3], aligned via ICP registration. After excluding shapes for which registration fails, the resulting dataset contains 42 categories with fully aligned images, shapes, and part annotations. Table 8 compares VLS and ZeroShape [3] combined with two point cloud segmentation methods: PointCLIPv2 [15] and Find3D [18].

Table 8: Quantitative Results on Objaverse.

Method	mIoU ↑	CD ↓
ZeroShape+PointCLIPv2	11.65	0.142
ZeroShape+Find3D	15.41	0.142
Ours	**19.70**	**0.130**

A.3 Quantitative Results on Generalization Analysis

State-of-the-art 2D open-vocabulary methods [75–81] typically leverage foundation models, such as CLIP [72] and SAM [61], pretrained on large-scale datasets covering diverse object categories. Consequently, categories designated as unseen in downstream evaluations may have already been encountered during foundation model pre-training. In contrast, the absence of comparably established 3D foundation models poses additional challenges for open-vocabulary 3D partonomic reconstruction. To address this limitation, we pretrained VLS on large-scale 3D shape datasets, including Objaverse [1] and ShapeNet [2], to learn generalizable 3D representations before fine-tuning on the downstream partonomic reconstruction task. To systematically evaluate the generalization capability of VLS, we conduct two complementary experiments examining generalization beyond pre-training categories and the transferability of part-level semantic knowledge.

Generalization Beyond Pre-Training Categories. We first exclude all unseen object categories from pre-training. The results obtained on PartNet [14, 31] are reported in Table 9. Excluding the unseen categories from pre-training reduces their mIoU from 35.89 % to 31.59 % and increases CD from 0.171 to 0.187. The degradation indicates that exposure to their unlabeled geometry and 3D CLIP embeddings during pre-training is beneficial. Nevertheless, VLS maintains strong performance even when these categories are excluded from pre-training. By contrast, performance on seen categories changes only marginally, from 39.41 % to 38.57 % (mIoU) and 0.200 to 0.209 (CD).

Table 9: Impact of Excluding Unseen Categories from Pre-Training.

Pre-Training Excludes Unseen Categories	Unseen Categories		Seen Categories	
	mIoU ↑	CD ↓	mIoU ↑	CD ↓
No	**35.89**	**0.171**	**39.41**	**0.200**
Yes	31.59	0.187	38.57	0.209

Generalization to Unseen Part Concepts. We further divide the unseen categories into two subsets: "Seen Part of Unseen Objects" and "Unseen Parts of Unseen Objects". The former evaluates whether VLS can transfer learned part concepts to novel object categories, while the latter considers the more challenging setting in which both object categories and part concepts are unseen during training. As shown in Table 10, VLS achieves better performance when part concepts remain familiar, demonstrating its ability to transfer part-level semantic knowledge across object categories. Furthermore, VLS outperforms ZeroShape [3] + Find3D [18] by 7.86 $\%$ on unseen parts of unseen objects, demonstrating stronger generalization to rare part concepts. This improvement highlights the effectiveness of our unified vision-language-shape representation and omni-supervised learning framework in handling previously unseen object parts.

Table 10: Generalization to Unseen Part Concepts (mIoU).

Method	Unseen Parts of Unseen Objects	Seen Parts of Unseen Objects	Unseen Objects Overall
Find3D	18.66	-	-
Ours	**26.52**	**35.56**	31.59

A.4 Quantitative Comparison with Partonomic 3D Reconstruction Model

We compare our proposed VLS with Ruan et al. [20] on PartNet. We conduct evaluation on their provided 5 categories (Bottle, Bowl, Display, Knife, and Mug). The results are reported in Table 11. It obviously shows VLS outperforms Ruan et al. [20] on both shape reconstruction (CD) and part decomposition (mIoU).

Beyond the quantitative gap, we highlight two structural advantages of VLS. First, Ruan et al. [20] train a separate model per category, requiring N models for N categories. VLS is a single model that generalizes across open-vocabulary categories without per-category retraining. Second, Ruan et al. [20] represent shape by deforming a template sphere mesh, which constrains outputs to a fixed mesh topology. VLS instead represent vision, language, and shape in a unified implicit field, allowing it to reconstruct shapes and parts of arbitrary topology (e.g., objects with holes or disconnected components).

Table 11: Quantitative Comparison with Partonomic 3D Reconstruction Model.

Method	mIoU ↑	CD ↓
Ruan et al. [20]	37.76	0.209
Ours	**42.07**	**0.172**

A.5 Quantitative Comparison with Part-Aware 3D Generation Model

We compare VLS with PartPacker [27], a current state-of-the-art part-aware generation method. Unlike our focus on *semantic* part decomposition, part-aware generation is *class-agnostic*: it predicts part masks on a shape without specifying or recognizing the parts' semantic categories. To evaluate the mIoU of PartPacker [27], we therefore match its class-agnostic masks to the ground truth semantic masks via Hungarian matching [91].

Chamfer Distance (CD) [87] is the most common reconstruction evaluation metric. Following ZeroShape [3], we additionally report F-score, which measures the point cloud alignment under a classification framing. Specifically, F-score@d (F@d) is the harmonic mean of precision@d and recall@d, where d is the distance threshold used to determine true and false positives. Other reconstruction metrics such as SSIM [92] and PSNR evaluate texture or appearance fidelity, which is not applicable for VLS.

The results on PartNet [14, 31] are reported in Table 12. VLS outperforms PartPacker [27] in both shape reconstruction and 3D part segmentation. It also outperforms the strongest two-model baseline, ZeroShape [3] + Find3D [18], under the F-score metric.

Table 12: Quantitative Comparison with Part-Aware 3D Generation Model.

Method	Unseen Categories					Seen Categories				
	mIoU ↑	CD ↓	F@1 ↑	F@2 ↑	F@5 ↑	mIoU ↑	CD ↓	F@1 ↑	F@2 ↑	F@5 ↑
PartPacker	24.07	0.275	23.22	50.37	82.75	28.58	0.409	17.41	42.98	74.78
ZeroShape+Find3D	26.72	0.210	28.35	54.28	86.53	23.48	0.289	22.26	49.44	81.89
Ours	**35.89**	**0.171**	**31.45**	**60.88**	**89.96**	**39.41**	**0.200**	**28.55**	**55.46**	**87.94**

A.6 More Ablation Results on 3D CLIP Loss

We further validate the 3D CLIP loss on the ShapeNetPart [30] dataset. The results are shown in Table 13. Consistent with the results on PartNet [14, 31], removing the 3D CLIP loss, i.e., the second term in Eq. 4, during pre-training degrades the shape CD. This supports the claim that language supervision improves shape reconstruction.

Table 13: Impact of 3D CLIP Loss on CD (ShapeNet).

	Unseen	Seen
w/o 3D CLIP Loss	0.288	0.331
Ours	**0.274**	**0.319**

A.7 Fine-Tuning Strategies

We compare LoRA [93] and head fine-tuning on PartNet [14, 31]. As shown in Table 14, LoRA [93] yields a marginal improvement in mIoU on unseen categories but performs slightly worse on the other three metrics: unseen CD, seen mIoU, and seen CD. Overall, the two fine-tuning strategies exhibit similar trade-offs between performance on seen and unseen categories.

Table 14: Quantitative Comparison between LoRA and Head Fine-Tuning.

Method	Unseen Categories		Seen Categories	
	mIoU ↑	CD ↓	mIoU ↑	CD ↓
LoRA [93]	**36.00**	0.178	39.15	0.208
Ours	35.89	**0.171**	**39.41**	**0.200**

A.8 Robustness to Camera Viewpoints

Because VLS is trained using input images of diverse viewpoints for each object, it is highly robust to variations in camera viewpoint. Table 15 reports its performance on PartNet from three different viewpoints.

A.9 Quantitative Comparisons with Weak-Perspective Projection

We use weak-perspective projection for two reasons. First, it avoids the need for known camera intrinsics, which are unavailable in our setting. Second, since the input objects are centered and scaled to a standard image size and reconstruction is performed within a unit cube, the effective focal length and object distance become normalized, which limits the impacts of weak-perspective approximation.

Specifically, we define the project matrix $\mathbf{K}$ so that the unit cube projects onto the full spatial extent of the visual feature map. The projection is used only to obtain an initial 2D reference location of each 3D query point. The deformable implicit function (DIFu) then adaptively adjusts these projected locations through learned offsets, allowing VLS to sample richer context of pixel embeddings and improve robustness to the weak-perspective approximation.

Nevertheless, weak-perspective projection has limitations, as it does not explicitly account for focal length or object distance and may introduce larger projection errors for objects with substantial depth variations. To investigate its impact, we provide an ablation study comparing weak-perspective

Table 15: Quantitative Comparison between Camera Viewpoints.

Viewpoint	mIoU ↑	CD ↓
Front	35.75	0.172
Side	**36.06**	**0.169**
Back	35.12	0.171

Table 16: Quantitative Comparison between Projection Methods.

	Unseen		Seen	
	mIoU ↑	CD ↓	mIoU ↑	CD ↓
Weak-Perspective Projection	35.89	0.171	39.41	0.200
Perspective Projection	**36.19**	**0.169**	**40.26**	**0.195**

projection with perspective projection computed using ground truth camera intrinsics. The results on PartNet [14, 31] are shown in Table 16. Perspective projection gives a small consistent improvement, which is expected given access to exact camera parameters, but the gap is modest. Because ground truth intrinsics are unavailable in our problem setting, weak-perspective projection remains a competitive choice.

A.10 Detailed Quantitative Results

Table 17 reports per-category results on unseen categories on ShapeNetPart [30] in terms of mIoU and CD. Table 18 further demonstrates per-category performance on unseen categories in PartNet with mIoU and CD. Overall, VLS achieves the best or second-best performance on most categories but does not perform as well on certain categories such as "Switch" and "Table". The aligned vision-language-shape representation, i.e., the lifted 3D CLIP embeddings, plays a crucial role in enabling generalizability of VLS to unseen categories. "Switch" is among the few categories whose parts are not well represented in this aligned space, as illustrated by the PCA visualization of the bottom-right example in Figure 7. In such cases, the semantic head has limited cues to locate parts, leading to inferior performance.

A.11 Applicability under Imperfect Textual Input

We further validate the applicability of our method under imperfect textual input, including long-form part descriptions and part names generated by a language model.

Long Descriptions. We replace short part names with long-form natural-language descriptions. Specifically, for each part, we use ChatGPT [71] to generate a one sentence description. Figure 9 presents qualitative results on unseen categories from ShapeNetPart [30]. The detailed long descriptions per category are listed in Table 19.

Unknown Part Names. We also evaluate a more practical setting in which users do not know the exact part vocabulary of an object. To simulate this scenario, we prompt ChatGPT [71] with "Please provide the coarse-level parts for [category]." We then use the returned part names as textual input to our model. Based on these approximate part names, our method reconstructs the object geometry and predicts its semantic part decomposition. Figure 10 shows qualitative results for three unseen categories in ShapeNetPart [30].

A.12 Additional Qualitative Results

Figure 11 provides additional qualitative results on unseen categories from ShapeNetPart [30]. The Figure 12 visualizes intermediate features using PCA before semantic embedding prediction. The left two examples correspond to seen categories, while the right two correspond to unseen categories.

Table 17: **Per-Category Quantitative Results for Unseen Categories on ShapeNetPart.**

Category	ZeroShape + PointCLIPV2		ZeroShape + PartSLIP++		ZeroShape + SATR		ZeroShape + ZeroPS		ZeroShape + Find3D		Ours	
	mIoU ↑	CD ↓	mIoU ↑	CD ↓	mIoU ↑	CD ↓	mIoU ↑	CD ↓	mIoU ↑	CD ↓	mIoU ↑	CD ↓
Bag	30.24	0.246	27.92	0.246	**47.83**	0.246	27.94	0.246	41.19	0.246	46.47	**0.228**
Earphone	20.49	**0.375**	18.47	**0.375**	13.75	**0.375**	18.60	**0.375**	**28.90**	**0.375**	25.40	0.401
Guitar	17.93	**0.300**	17.24	**0.300**	14.08	**0.300**	17.25	**0.300**	15.01	**0.300**	**22.65**	0.302
Rocket	10.25	0.147	16.36	0.147	21.96	0.147	16.29	0.147	22.49	0.147	**26.15**	**0.138**
Table	24.84	0.319	20.65	0.319	21.12	0.319	20.66	0.319	**32.44**	0.319	23.69	**0.303**
Average	20.75	0.277	20.13	0.277	23.78	0.277	20.15	0.277	25.56	0.277	**28.87**	**0.274**

Table 18: **Per-Category Quantitative Results for Unseen Categories on PartNet.**

Category	ZeroShape + PointCLIPv2		ZeroShape + PartSLIP++		ZeroShape + SATR		ZeroShape + ZeroPS		ZeroShape + Find3D		Ours	
	mIoU ↑	CD ↓	mIoU ↑	CD ↓	mIoU ↑	CD ↓	mIoU ↑	CD ↓	mIoU ↑	CD ↓	mIoU ↑	CD ↓
Bucket	11.90	0.213	17.66	0.213	17.44	0.213	17.71	0.213	23.68	0.213	**44.62**	**0.198**
DishWasher	16.76	0.250	24.30	0.250	32.92	0.250	24.29	0.250	33.39	0.250	**60.10**	**0.135**
Dispenser	12.27	0.194	10.01	0.194	2.66	0.194	10.03	0.194	16.15	0.194	**12.45**	**0.154**
FoldingChair	23.07	0.310	17.54	0.310	21.19	0.310	17.54	0.310	19.14	0.310	**26.76**	**0.234**
KitchenPot	17.07	0.171	17.95	0.171	11.36	0.171	18.00	0.171	20.70	0.171	**40.78**	**0.153**
Pot	14.99	0.145	31.86	0.145	25.23	0.145	31.91	0.145	53.57	0.145	**69.89**	**0.144**
Refrigerator	17.09	0.142	17.81	0.142	6.77	0.142	17.80	0.142	19.04	0.142	**29.58**	**0.132**
Remote	13.10	0.163	16.03	0.163	25.13	0.163	16.08	0.163	25.29	0.163	**46.84**	**0.125**
Switch	21.07	0.215	28.37	0.215	21.22	0.215	28.34	0.215	**43.10**	0.215	10.99	**0.205**
Table	8.01	0.291	18.52	0.291	18.48	0.291	18.47	0.291	**26.04**	0.291	19.89	**0.232**
Window	16.63	0.215	15.35	0.215	20.22	0.215	16.95	0.215	13.78	0.215	**32.84**	**0.167**
Average	15.63	0.210	19.57	0.210	18.42	0.210	19.74	0.210	26.72	0.210	**35.89**	**0.171**

A.13 Limitations

We acknowledge that the qualitative comparisons for table do not fully reflect the quantitative results, and we provide more diverse and representative examples in Figure 13. Specifically, we identify two primary types of failure.

The first is **semantic-representation failure**, where the overall geometry is reconstructed reasonably well, but the learned semantic field fails to distinguish individual parts accurately. As shown in Table 18, VLS achieves better shape CD than the baselines for the Switch and Table categories, yet obtains relatively low part mIoU. In particular, VLS achieves a CD of 0.205 but only 10.99% mIoU on Switch. As illustrated in Figure 7, the lifted 3D CLIP embeddings for Switch lack spatially distinct semantic regions, providing insufficient cues for accurate part localization. This highlights the dependence of VLS on the quality of the aligned vision-language-shape representation.

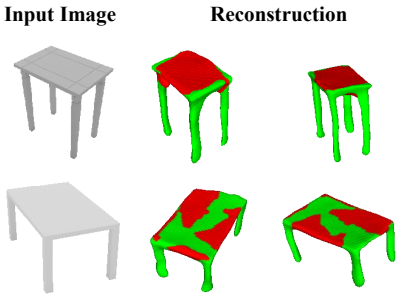

Figure 13: Visualization of Failure Cases in Table.

The second is **coupled geometry-and-semantic failure**, where inaccurate shape reconstruction further degrades semantic part decomposition. For example, on the Earphone category in ShapeNetPart [30], VLS achieves both worse CD and lower mIoU than the baseline. Earphone contain thin and disconnected structures, such as headbands, earcups, and wires, which are challenging to reconstruct accurately. Errors in reconstructing these structures consequently affect part localization, leading to degraded performance in both geometry and semantics.

Furthermore, our current framework focuses on a single, flat level of part granularity for each object category. Extending VLS to fine-grained and hierarchical partonomic reconstruction remains an important direction for future work, requiring explicit modeling of parent-child relationships, cross-level consistency, and overlapping part representations at different granularities.

Table 19: Part Descriptions Generated by ChatGPT for Object Categories in Figure 9.

Category	Part Name	Description Generated by ChatGPT
Bag	Body	The main storage compartment that holds the contents.
	Handle	The short grip used to carry the bag by hand.
Guitar	Neck	A long, narrow part that connects the body to the head and supports the fretboard.
	Body	The large main section that amplifies sound and supports the other guitar parts.
	Head	The end section that holds the tuning pegs for adjusting string tension.
Table	Top	The flat upper surface used to place objects or perform activities.
	Legs	The supporting vertical parts that raise and hold the tabletop above the ground.
	Drawer	A sliding storage compartment built into the table for holding small items.

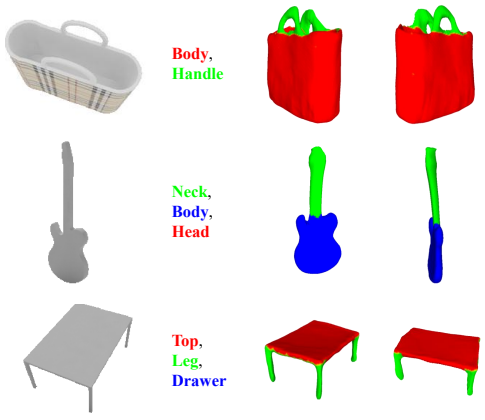

Figure 9: Qualitative Results using Long-Form Part Descriptions in Table 19 on ShapeNetPart.

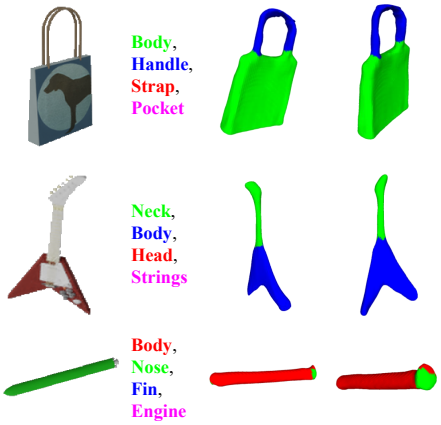

Figure 10: Qualitative Results using LLM-Generated Part Names on ShapeNetPart.

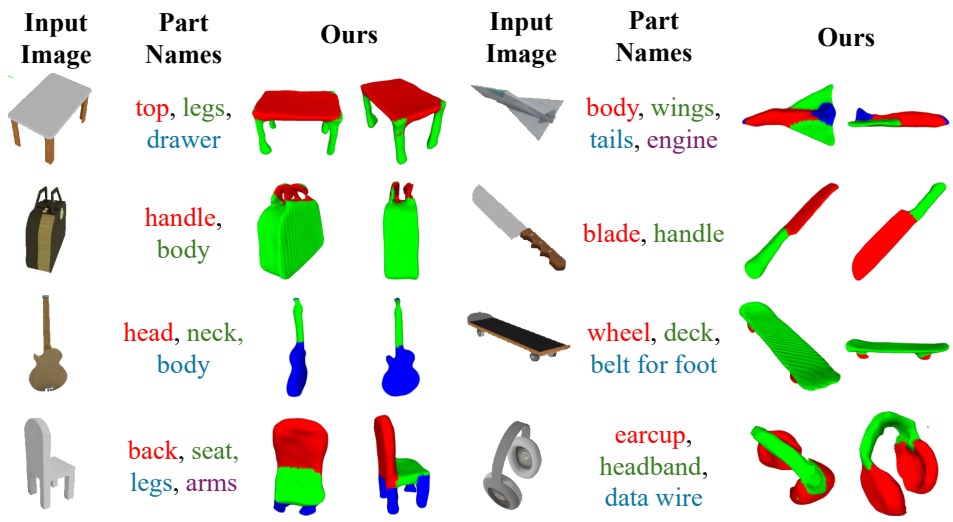

Figure 11: Additional Qualitative Examples for Unseen Categories in ShapeNetPart.

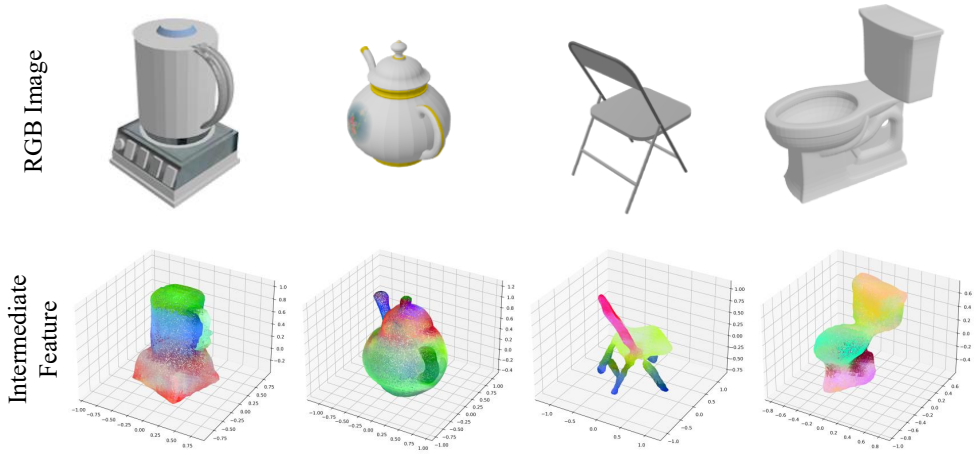

Figure 12: PCA Visualization of Intermediate Features before Semantic Embedding Prediction. The left two examples correspond to seen categories, while the right two correspond to unseen categories.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The contributions and scope of open-vocabulary partonomic 3D reconstruction are discussed in Abstract and Introduction.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: Refer to Appendix.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [N/A]

Justification: This paper focuses on a practical, application-driven framework and does not present formal theoretical results requiring assumptions or proofs.

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: Refer to Section 4 and Appendix.

   Guidelines:

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

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

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

Answer: [Yes]

Justification: The code and data have been released.

Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: Refer to Section 4 and Appendix.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: Refer to Section 4.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: Refer to Section 4.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: Yes, we reviewed the Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: Refer to Section 5.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [N/A]

    Justification: This paper poses no such risks.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: We use existing public dataset, models, and code and properly cite the corresponding papers.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [Yes]

    Justification: This paper introduces new assets, including the processed dataset and trained model weights, which will be released after paper acceptance.

    Guidelines:

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

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

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

    Answer: [N/A]

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

    Guidelines:

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

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

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

    Answer: [N/A]

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

    Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [N/A]

Justification: LLMs was not used to core method development.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.