UNIVERSITY OF CALIFORNIA SAN DIEGO

Open-World 3D Understanding and Generation

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy

in

Computer Science

by

Minghua Liu

Committee in charge:

Professor Hao Su, Chair
Professor Ravi Ramamoorthi
Professor Zhuowen Tu
Professor Rose Yu

2024

The Dissertation of Minghua Liu is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2024

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I want to express my deepest gratitude to my advisor, Professor Hao Su, whose guidance and support have been instrumental in shaping not only this dissertation but also my growth as a researcher and individual. I have been continually amazed by Professor Su's admirable courage in embracing uncertainty and venturing beyond his comfort zone. His foray into robotics exemplifies his dedication to pushing the boundaries of fundamental problems. His willingness to learn alongside his PhD students and his bold step into entrepreneurship demonstrate a remarkable commitment to both personal and professional growth. This fearlessness in the face of new challenges, coupled with his ethos of continuous learning, has taught me the value of intellectual bravery and adaptability in research.

I am particularly grateful for Professor Su's role in honing my research acumen. His guidance in developing a discerning research taste and identifying fundamental problems has been invaluable. His pure curiosity and relentless pursuit of truth have set a profound example for me. His insightful vision not only illuminates the path of our field but also extends to the broader world. Beyond his academic brilliance, Professor Su's personal qualities have made this journey truly enriching. His tolerance, approachability, and selflessness have created an environment conducive to open dialogue and intellectual exploration. His genuine willingness to support young researchers like myself has been a cornerstone of my doctoral experience.

I would like to extend my heartfelt thanks to my labmates (listed alphabetically): Fangchen Liu, Fanbo Xiang, Isabella Liu, Jiayuan Gu, Mukund Varma T, Quan Vuong, Ruoxi Shi, Songfang Han, Stone Tao, Tongzhou Mu, Xiaodi Yuan, Xiaoshuai Zhang, Xinyue Wei, Xuanlin Li, Yuchen Zhou, Yulin Liu, Yunhao Fang, Yuzhe Qin, Zhan Ling, and Zhiao Huang. You are more than just colleagues; you are friends who have made this journey both enjoyable and rewarding. I deeply appreciate your knowledge, helpfulness, and self-motivation, as well as the inspiring and vibrant environment we have fostered together.

I am incredibly fortunate to have collaborated with many brilliant coauthors (listed alphabetically): Ang Li, Boqing Gong, Charles R. Qi, Dragomir Anguelov, Fatih Porikli,

Hansheng Chen, Hong Cai, Kaiming Kuang, Mengqi Zhang, Minhyuk Sung, Radomir Mech, Shizhong Han, Yangyan Li, Yin Zhou, Yinhao Zhu, Zexiang Xu, Zhaoning Wang, Zhuowen Tu, and Zhuoyang Zhang. Special thanks are due to Chao Xu, Chong Zeng, Haian Jin, and Linghao Chen. Your expertise, insights, dedication, and collaboration have significantly enriched this journey. I thoroughly enjoyed the collaborations and learned a great deal from each of you.

Outside of research, I am incredibly fortunate to be surrounded by supportive friends throughout my PhD journey, including (listed alphabetically): An Yan, Canwen Xu, Chenyang An, Chengyu Dong, Enze Liu, Hanxian Huang, Jiarui Xu, Jiayun Zhang, Jun Yan, Ke Chen, Li Zhong, Qinyuan Ye, Ruihan Yang, Shihan Ran, Shuheng Li, Tiange Luo, Yuanchen Guo, Yuheng Zhi, Zexue He, Zhankui He, Zi Lin, and Zilong Wang. Their companionship, support, and encouragement have been invaluable to me, and I am deeply grateful for their presence in my life.

Lastly, I want to express my deepest gratitude to those who have been my pillars of strength throughout this journey. To my parents: your unwavering support, unconditional love, and steadfast belief in me have shaped every step of this path. The early habits you subtly instilled and your trust in all my decisions have been the foundation upon which this work has been built. To my girlfriend Xiyuan Zhang: your love and companionship have filled my life with joy, and I am deeply grateful for your presence through both the highs and lows of this journey. Your patience, emotional steadiness, and faith in me have been constant sources of encouragement. I am especially thankful for your understanding and for the many memories we've created together. A special thank you to Jojo, my loyal four-legged companion, whose joyful presence and boundless energy have brought comfort and balance to my life, especially during the most challenging times. Finally, a nod to the beautiful San Diego and California weather, which provided the perfect backdrop for reflection, creativity, and much-needed breaks. Each of you has played an indispensable role in helping me reach this milestone, and for that, I am eternally grateful.

Chapter 2 incorporates material from the publication "One-2-3-45: Any Single Image to

3D Mesh in 45 Seconds without Per-Shape Optimization", by Minghua Liu, Chao Xu, Haian Jin, Linghao Chen, Mukund Varma T, Zexiang Xu, and Hao Su, published in *Advances in Neural Information Processing Systems (NeurIPS 2023)*. The dissertation author was primary investigator and the lead author of this paper.

Chapter 3 incorporates material from the publication "One-2-3-45++: Fast Single Image to 3D Objects with Consistent Multi-View Generation and 3D Diffusion", by Minghua Liu, Ruoxi Shi, Linghao Chen, Zhuoyang Zhang, Chao Xu, Xinyue Wei, Hansheng Chen, Chong Zeng, Jiayuan Gu, and Hao Su, published in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2024)*. The dissertation author was primary investigator and the lead author of this paper.

Chapter 4 has been partially submitted for peer review as the paper titled "MeshFormer: High-Quality Mesh Generation with a 3D-Guided Reconstruction Model", authored by Minghua Liu, Chong Zeng, Xinyue Wei, Ruoxi Shi, Linghao Chen, Chao Xu, Mengqi Zhang, Zhaoning Wang, Xiaoshuai Zhang, Isabella Liu, Hongzhi Wu, and Hao Su. The dissertation author was the primary investigator and lead author of this paper.

Chapter 5 incorporates material from the publication "OpenShape: Scaling Up 3D Shape Representation Towards Open-World Understanding", by Minghua Liu, Ruoxi Shi, Kaiming Kuang, Yinhao Zhu, Xuanlin Li, Shizhong Han, Hong Cai, Fatih Porikli, and Hao Su, published in *Advances in Neural Information Processing Systems (NeurIPS 2023)*. The dissertation author was primary investigator and the lead author of this paper.

Chapter 6 incorporates material from the publication "PartSLIP: Low-Shot Part Segmentation for 3D Point Clouds via Pretrained Image-Language Models", by Minghua Liu, Yinhao Zhu, Hong Cai, Shizhong Han, Zhan Ling, Fatih Porikli, and Hao Su, published in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2023)*. The dissertation author was primary investigator and the lead author of this paper.

VITA

2019	Bachelor of Engineering in Computer Science and Technology, Tsinghua University
2024	Doctor of Philosophy in Computer Science, University of California San Diego

PUBLICATIONS

Minghua Liu[*], Chong Zeng[*], Xinyue Wei, Ruoxi Shi, Linghao Chen, Chao Xu, Mengqi Zhang, Zhaoning Wang, Xiaoshuai Zhang, Isabella Liu, Hongzhi Wu, and Hao Su. MeshFormer : High-Quality Mesh Generation with 3D-Guided Reconstruction Model. Under review.

Seonghun Oh, Xiaodi Yuan, Xinyue Wei, Ruoxi Shi, Fanbo Xiang, **Minghua Liu**, and Hao Su. PaMO: Parallel Mesh Optimization for Intersection-Free Low-Poly Modeling on the GPU. Under review. 2024.

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 (ECCV)*. 2024.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2024.

Tongzhou Mu, **Minghua Liu**, and Hao Su. Learning Reusable Dense Rewards for Multi-Stage Tasks. In *International Conference on Learning Representations (ICLR)* 2024.

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. In *Advances in Neural Information Processing Systems (NeurIPS)*. 2023.

Minghua Liu[*], Ruoxi Shi[*], Kaiming Kuang[*], Yinhao Zhu, Xuanlin Li, Shizhong Han, Hong Cai, Fatih Porikli, and Hao Su. OpenShape: Scaling Up 3D Shape Representation Towards Open-World Understanding. In *Advances in Neural Information Processing Systems (NeurIPS)*. 2023.

Xuanlin Li[*], Yunhao Fang[*], **Minghua Liu**, Zhan Ling, Zhuowen Tu, and Hao Su. Distilling Large Vision-Language Model with Out-of-Distribution Generalizability. In *Proceedings of the*

[*] indicates equal contribution, and [†] indicates equal advising.

International Conference on Computer Vision (ICCV). 2023.

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 (CVPR)*. 2023.

Xiaoshuai Zhang*, Rui Chen*, Fanbo Xiang**, Yuzhe Qin**, Jiayuan Gu**, Zhan Ling**, **Minghua Liu****, Peiyu Zeng**, Songfang Han***, Zhiao Huang***, Tongzhou Mu***, Jing Xu, and Hao Su. Close the Visual Domain Gap by Physics-Grounded Active Stereovision Depth Sensor Simulation. In *IEEE Transactions on Robotics (T-RO)*. 2023.

Minghua Liu*, Xuanlin Li*, Zhan Ling*, Yangyan Li, and Hao Su. Frame Mining: a Free Lunch for Learning Robotic Manipulation from 3D Point Clouds. In *Conference on Robot Learning (CoRL)*. 2022.

Minghua Liu, Yin Zhou, Charles R. Qi, Boqing Gong, Hao Su, and Dragomir Anguelov. LESS: Label-Efficient Semantic Segmentation for LiDAR Point Clouds. In *European Conference on Computer Vision (ECCV)*. 2022.

Xinyue Wei*, **Minghua Liu***, Zhan Ling, and Hao Su. Approximate Convex Decomposition for 3D Meshes with Collision-Aware Concavity and Tree Search. In *ACM Transactions on Graphics (TOG)*. 2022.

Minghua Liu, Minhyuk Sung, Radomir Mech, and Hao Su. DeepMetaHandles: Learning Deformation Meta-Handles of 3D Meshes with Biharmonic Coordinates. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021.

Jiachen Li, Quan Vuong, Shuang Liu, **Minghua Liu**, Kamil Ciosek, Henrik Christensen, and Hao Su. Multi-task Batch Reinforcement Learning with Metric Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*. 2020.

Minghua Liu, Xiaoshuai Zhang, and Hao Su. Meshing Point Clouds with Predicted Intrinsic-Extrinsic Ratio Guidance. In *European Conference on Computer Vision (ECCV)*. 2020.

Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, **Minghua Liu**, Hanxiao Jiang, Yifu Yuan, He Wang, Li Yi, Angel X Chang, Leonidas J Guibas, and Hao Su. SAPIEN: A Simulated Part-Based Interactive Environment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020.

Minghua Liu, Lu Sheng, Sheng Yang, Jing Shao, and Shi-Min Hu. Morphing and Sampling Network for Dense Point Cloud Completion. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*. 2020.

Minghua Liu, Hang Ma, Jiaoyang Li, and Sven Koenig. Task and Path Planning for Multi-Agent Pickup and Delivery. In *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*. 2019.

Sheng Yang, Beichen Li, **Minghua Liu**, Yu-Kun Lai, Leif Kobbelt, and Shi-Min Hu. HeteroFusion: Robust Scene Reconstruction using Heterogeneous Sensors on Robots. In *IEEE Transactions on Visualization and Computer Graphics (TVCG)*. 2019.

Sheng Yang, Kang Chen, **Minghua Liu**, Hongbo Fu, and Shi-Min Hu. Saliency-Aware Real-Time Volumetric Fusion for Object Reconstruction. In *Computer Graphics Forum*. 2017.

ABSTRACT OF THE DISSERTATION

Open-World 3D Understanding and Generation

by

Minghua Liu

Doctor of Philosophy in Computer Science

University of California San Diego, 2024

Professor Hao Su, Chair

3D representations model our physical world in one of the most explicit and structured ways, enabling the storage of extensive attributes. Understanding these 3D representations—including, but not limited to, their geometry, appearance, structure, semantics, mobility, functionality, and affordances—is crucial for developing intelligent agents that can comprehend and interact with our 3D physical environment and seamlessly integrate into human settings. Additionally, high-quality 3D generation allows us to replicate our 3D world, creating digital twins and supporting a wide range of downstream applications. Significant advancements have already been made in 3D deep learning by exploring suitable representations and neural algorithms for 3D data. However, unlike many other modalities, the scale of publicly available 3D data has

been quite limited. Most previous 3D deep learning approaches have traditionally been confined to a narrow range of common categories (such as chairs, cars, airplanes, etc.), which greatly hinders their application to real-world scenarios with far more diverse categories and variations. Nevertheless, in the past two years, with the rapid development of large-scale pretrained models from 2D vision, language, and other modalities, as well as the emergence of larger and more diverse public 3D datasets, many new opportunities for 3D deep learning have arisen.

In this dissertation, we explore how to leverage extensive priors from other modalities, as well as how to exploit the limited but valuable 3D data to enhance the generalizability of various 3D deep learning tasks. We primarily focus on two families of tasks: 3D object understanding and 3D object generation in an open-world context. For each family, I explore several strategies to effectively utilize these priors and identify a series of crucial problems with proposed solutions. My efforts have significantly improved the generalizability of many previous 3D understanding and generation tasks, bridging the gap between the capabilities of earlier 'chair research' and the complex, diverse open-world scenarios in the real physical 3D world.

Chapter 1

Introduction

1.1 Why Do We Need 3D Understanding and Generation?

As we advance toward creating Artificial General Intelligence (AGI) and developing intelligent agents that can seamlessly integrate into human environments, the need for comprehensive 3D understanding and generation becomes increasingly critical.

AGI involves developing machines that possess the cognitive abilities to perform any intellectual task that a human can. Achieving AGI requires systems that can perceive and interact with the world in a manner akin to humans. While significant progress has been made in artificial intelligence, particularly in language models [2, 57], these advancements have largely focused on virtual tasks, knowledge application, and abstract reasoning. However, the ultimate goal of building intelligent agents extends beyond merely providing knowledge and suggestions. We envision agents that can assist us with practical, everyday tasks—agents capable of performing chores, cooking dinner, and walking the dog. For instance, while an LLM can provide detailed instructions on how to assemble furniture, it lacks the physical grounding to understand the spatial relationships between the parts or to interact directly with the objects. For an intelligent agent to operate in the real world, it must be capable of understanding spatial relationships, navigating complex environments, and interacting with physical entities, all of which require a deep understanding of 3D spaces.

Humans live in a three-dimensional world. While 2D images and videos are valuable tools for capturing and conveying visual information, they are inherently limited in their ability

to represent the full complexity of the physical world. These representations lack the depth, scale, and spatial relationships critical for understanding and interacting with three-dimensional spaces. In contrast, 3D representations provide a structured and explicit way to model the physical world, capturing not only low-level features such as geometry and appearance but also high-level information such as semantics, structure, mobility, and affordances. This richness of structure and information is crucial for tasks that require accurate perception, reasoning, and interaction within a 3D environment.

In addition to perceiving and understanding the physical world, 3D generation enables us to create accurate digital representations—or "digital twins"—of objects and environments. These digital twins are virtual replicas of physical entities, capturing their appearance, structure, and physical properties in a 3D format. This capability is critical not only for practical applications in robotics and automation but also for broader domains like simulation [334, 385], the metaverse, and virtual reality. Specifically, digital twins serve as the foundation for simulations, allowing researchers and engineers to model, test, and optimize real-world processes in a controlled virtual environment [140, 159, 210]. This ability to run simulations on digital twins reduces costs, minimizes risks, and accelerates the development process, making it an invaluable tool in various industries. The benefits of 3D generation also extend into the realm of the metaverse—a collective virtual shared space that merges physical and digital realities. In the metaverse, 3D generation plays a crucial role in creating immersive environments where users can interact with digital representations of real-world objects and spaces, providing experiences that closely mirror the physical world. This opens up new possibilities for entertainment, social interaction, education, and commerce, where the boundaries between the real and virtual worlds become increasingly blurred.

1.2 Advancements in 3D Deep Learning: Representations and Algorithms

Traditionally, 3D computing has been an interdisciplinary endeavor, intertwining concepts from geometry and topology in mathematics with computer vision (CV), computer graphics (CG), robotics, and computer-aided design (CAD) in computer science. Before the advent of deep learning, most research in this field focused on the analysis or synthesis of individual or small collections of 3D objects or scenes [87, 98]. These classical approaches, while groundbreaking at the time, faced significant limitations in scalability, adaptability, and efficiency, prompting the need for more robust and generalized methods. Although there have been efforts in classical data-driven 3D understanding algorithms [45, 120], these typically required intricate feature engineering and handcrafted models, which were both time-consuming and limited in their ability to generalize across diverse datasets.

In response to these challenges, the field has witnessed a paradigm shift with the advent of deep learning. Deep learning methods, characterized by their ability to automatically learn features and representations from data, have introduced a unified framework for solving a wide range of data understanding problems. This shift has not only streamlined the development process but also opened up new possibilities for tackling complex 3D data analysis and generation tasks. Over the past few years, research in 3D deep learning has made significant advancements, mainly centered around three key problems:

3D Representations for Deep Learning A key aspect of 3D deep learning research revolves around the representation of 3D data. Unlike images, text, and videos, which benefit from uniform and well-established representations, 3D data lacks a universally efficient representation. The naive extension of 2D grids to 3D volumes, while conceptually straightforward and compatible with convolutional neural network (CNN) architectures, suffers from cubic complexity, making it computationally expensive and impractical for many applications. Consequently, researchers have explored various alternative representations to better suit the needs of neural networks.

For example, point clouds represent 3D objects as a collection of discrete points in space, offering a more compact and flexible alternative to volumetric grids. Another promising representation is the mesh, which encodes the surface geometry of a 3D object as a collection of vertices, edges, and faces. Meshes provide a more detailed and accurate representation of an object's surface compared to point clouds or volumes, making them ideal for tasks such as 3D reconstruction and shape analysis. More recently, implicit field representations have emerged as a powerful alternative for representing 3D data. These representations, including Neural Radiance Fields (NeRF) [202] and Signed Distance Fields (SDF), model 3D shapes as continuous functions that map spatial coordinates to scalar values, such as occupancy or distance to the object's surface. Implicit fields offer several advantages, including the ability to represent complex shapes with high fidelity and the flexibility to handle varying levels of detail.

Learning Algorithms and Network Architectures for 3D Analysis The design of learning algorithms and networks for 3D data is closely tied to the 3D representation used. Since many 3D representations are irregular or possess high complexity, it is non-trivial to design neural networks that can effectively process and analyze 3D data. Over the past decade, a wide variety of network architectures have been proposed to tackle the challenges posed by different 3D representations.

For instance, 3D convolutional neural networks (3D CNNs) have become the standard for volumetric data, with subsequent work focused on improving their efficiency and scalability. For point cloud data, specialized architectures such as PointNet [228] and its variants [229, 231] have been developed, which directly operate on point sets without the need for voxelization. These architectures leverage permutation invariance to extract meaningful features from unordered point clouds. For mesh-based data, graph neural networks (GNNs) have gained prominence due to their ability to operate on non-Euclidean data structures and meshes can be naturally represented as graphs.

Network Architectures and Algorithms for 3D Generation 3D deep learning also involves the generation of 3D data, which presents unique challenges compared to traditional tasks

such as classification or regression. Generating a 3D representation requires the network to produce a complex and potentially irregular output, such as an occupancy volume [199], point cloud [62], mesh [213], or implicit field [222]. Various strategies have been proposed to address this challenge, each tailored to the specific representation being used.

For instance, 3D (sparse) convolutional networks [276] can be utilized for generating occupancy or SDF volumes. Some researchers have explored generating point sets by directly outputting point sequences or deforming 2D grids [78]. Others have investigated leveraging transformer models used for sequence generation [32, 213, 265], for the generation of polygon meshes. More recently, there has been a focus on representing implicit fields, either by directly using multi-layer perceptrons or by exploring data structures such as triplanes, hash grids, feature volumes, and various decomposition strategies.

1.3 Challenges in 3D Deep Learning: Data Scarcity

As discussed in Section 1.2, the field of 3D deep learning has witnessed remarkable advancements in recent years, with various research efforts focused on identifying the most suitable 3D representations for deep learning, alongside the development of innovative algorithms and architectures tailored to these representations. Despite these strides, the challenge of data scarcity remains a persistent and significant hurdle in the advancement of 3D deep learning. This issue of limited data availability is particularly troubling when compared to the abundance of data in other domains such as text and images.

Complexity in 3D Data Collection Unlike text and image data, one of the primary reasons for the scarcity of 3D data is the inherent complexity involved in its collection. A significant portion of existing 3D data is manually constructed by skilled artists using professional software tools like Blender or Maya. The creation of high-quality 3D models is a labor-intensive process that can take anywhere from several hours to days, depending on the complexity and level of detail required. This process involves intricate tasks such as modeling, texturing, rigging, and

rendering, all of which demand a high level of expertise and precision. As a result, the generation of 3D data is not only time-consuming but also costly, limiting the availability of large-scale datasets.

Additionally, the use of 3D sensors, which could potentially automate the data collection process, is not yet widespread. While sensors like LiDAR and structured light scanners are capable of capturing 3D data, their adoption is still limited to specific industries such as autonomous driving and industrial inspection. These sensors are expensive, and are often limited in their ability to capture complex environments. Furthermore, 3D reconstruction techniques, such as Structure from Motion (SfM), while promising, are also sensitive to noise and occlusions, making it challenging to produce high-quality 3D models in uncontrolled environments [359, 360]. Consequently, the collection of 3D data remains a non-trivial task, further exacerbating the issue of data scarcity.

Limitations of Public 3D Datasets As a result, existing public 3D datasets are often limited in scale. One of the most widely used 3D shape datasets, ShapeNet Core [22], contains only around 51,300 shapes covering 55 object categories. While this dataset has been instrumental in advancing research in 3D shape analysis and generation, it falls short in terms of the diversity and scale required for developing robust and generalizable 3D models. The limited object diversity in datasets like ShapeNet Core is a significant concern, particularly in the context of open-world 3D generation and understanding, where an intelligent agent is expected to encounter a vast array of objects in real-world applications. The paucity of diverse 3D data restricts the ability of models to generalize to unseen object categories, limiting their applicability in real-world scenarios.

Complexity in 3D Data Annotation The challenge of limited dataset scale is further amplified when it comes to fine-grained annotations. Annotating 3D data is a far more complex and resource-intensive task compared to annotating 2D images. For instance, in image segmentation, annotators can use simple tools like rectangles or brushes to label regions of interest. In contrast, annotating 3D data involves more intricate operations such as rotating, translating, and scaling the mesh to accurately identify and label specific regions. This complexity makes it difficult

to scale the annotation process, resulting in a scarcity of fine-grained annotated datasets. The PartNet dataset [206], one of the largest datasets containing fine-grained part labels for 3D shapes, includes only 16 object categories, highlighting the challenges of producing detailed annotations at scale. The scarcity of such datasets severely limits the ability of 3D deep learning models to perform tasks that require detailed understanding of object parts and their relationships.

Impact on 3D-Supervised 3D Deep Learning The scarcity of 3D data has profound implications for the performance and generalizability of 3D deep learning models. Traditional 3D deep learning algorithms have largely followed a supervised training paradigm, where the performance of a model is heavily dependent on the quality and quantity of the training data. In the context of 3D data, the limited availability of diverse and annotated datasets results in models that suffer from poor generalization. These models are often trained on a narrow set of object categories, such as chairs, planes, and cars, and struggle to perform well on unseen categories or in complex, real-world environments.

For analysis tasks such as classification, detection, and segmentation, the lack of diverse training data means that models are likely to overfit to the limited categories present in the dataset, resulting in reduced accuracy and robustness when applied to new data. Similarly, for generative tasks, such as 3D object generation and scene synthesis, the scarcity of data limits the model's ability to produce realistic and diverse outputs. In open-world scenarios, where an intelligent agent is expected to generate and understand a wide range of objects and environments, this limitation poses a significant challenge.

1.4 From "Chair Research" to Open World: New Opportunities

As discussed in Section 1.3, many early approaches in 3D deep learning followed the conventional supervised learning paradigm. While this framework proved effective in domains with abundant data, such as image and text, it struggled in the 3D domain due to the limited

availability of publicly accessible datasets. As a result, these methods were often restricted to a narrow range of object categories, with "chair" being the most frequently used category for experiments and demonstrations. This reliance on a limited set of categories significantly hindered the generalizability of these methods, making it challenging to extend their applicability to open-world 3D understanding and generation.

However, the landscape of 3D deep learning has undergone a transformative shift over the past two years. The confluence of advancements in large-scale pre-trained models and the emergence of more extensive and diverse 3D datasets has opened up new opportunities, transitioning the field from the so-called "chair research" to the more ambitious goal of open-world 3D generation and understanding.

The Power of Large-Scale Pre-trained 2D/Text Models One of the critical factors driving this transition has been the success of large-scale pre-trained models in other domains, particularly in 2D image and natural language processing. Unlike the 3D domain, where data has historically been scarce, images and text are abundant on the internet. By leveraging vast datasets such as LAION-5B [256], which includes over five billion image-text pairs, researchers have been able to train scalable models like large-scale transformers and diffusion models. These models, such as Stable Diffusion [247] and DALL-E [237] for 2D image generation, large language models like ChatGPT [217] for text generation, and vision-language models like CLIP and Flamingo for multi-modal interaction, have demonstrated remarkable capabilities in zero-shot learning and open-world tasks. By pretraining on diverse and extensive image-text pairs, these models learn rich visual concepts and knowledge, enabling them to perform a wide range of novel tasks that were previously unattainable.

The Emergence of Large-Scale 3D Datasets In parallel with advancements in pre-trained models, the 3D domain has witnessed the emergence of new datasets that significantly expand the scope and diversity of available 3D data. Previously, datasets like ShapeNet [22], which contained only tens of object categories, were the primary resources for training 3D models. However, recent efforts to crawl and aggregate data from sources like Sketchfab have resulted

in the creation of much larger datasets, such as Objaverse [50] and Objaverse-XL [49]. The scale of these datasets has grown from 51,000 objects in earlier collections to 800,000 in Objaverse and over 10 million in Objaverse-XL. While these numbers are still several orders of magnitude smaller than those in the image and text domains, the increase in diversity is significant. Objaverse, for example, includes over a thousand object categories, offering a much richer and more varied dataset for training and evaluating 3D models.

New Opportunities in Open-World 3D Research The combination of larger and more diverse datasets with advanced pre-trained models offers unprecedented opportunities to extend the scope of 3D research beyond the limited "chair research" of the past. These new resources provide a more extensive prior knowledge base for tackling various 3D tasks, both in understanding and generation. Moreover, the natural connection between 2D and 3D data opens up exciting possibilities for leveraging the successes of 2D pre-trained models to enhance 3D deep learning.

The key challenge now lies in effectively extracting, distilling, and leveraging the knowledge encoded in 2D pre-trained models and fully exploiting the larger and more diverse 3D datasets, despite the continued limitations in the scale of 3D data. By building on the successes of 2D and text pre-trained models, and by maximizing the potential of growing 3D datasets, researchers can push the boundaries of what is possible in 3D deep learning. The transition from "chair research" to open-world 3D understanding and generation represents not only a shift in focus but also a significant step forward in the capabilities of 3D deep learning, opening up new avenues for research and application.

1.5 Overview of Methods and Contributions

In this dissertation, we explore how to leverage extensive priors from other modalities, such as image or text pre-trained models, while also exploiting the limited but valuable 3D data to enhance the generalizability of various 3D deep learning tasks. We primarily focus on two families of tasks: 3D object generation (Chapters 2, 3 and 4) and 3D object understanding

(Chapters 5 and 6) within an open-world context. For each family, I discuss several strategies to effectively utilize these priors and address a series of crucial problems. Please refer to Figure 1.1 for an overview of our contribution.contribution.

1.5.1 Open-World 3D Generation

Unlike 3D reconstruction, which typically benefits from "complete" information provided by dense views, 3D generation from a single-view image or text prompt requires more extensive "hallucination" for invisible and occluded regions. As a result, these tasks demand more comprehensive priors. While many prior works have studied the design of generative models for various 3D representations, these models usually generate only a few object categories, such as chairs, cars, and planes, and struggle with open-world generation. To achieve open-world 3D generation, there are two main strategies for leveraging priors from 2D pretrained models.

The first strategy involves projecting 3D to 2D multi-view images and leveraging 2D models to calculate a loss or gradient for guidance. This paradigm, pioneered by DreamFusion [225], optimizes a 3D representation (e.g., NeRF) for each input text or image and then uses 2D diffusion models to calculate a score distillation sampling (SDS) loss to guide the optimization of the 3D representation. While capable of handling open-world 3D generation, these approaches typically require hours to generate a single 3D shape and suffer from several issues, such as the Janus problem (e.g., generating multiple heads), over-saturation, and poor geometry.

Another strategy is to fine-tune 2D diffusion models to extend their capability by adding various conditions and generating the 2D images that we want. For example, a small amount of 3D data can be used to stimulate 2D diffusion models to generate novel views of the object of interest. In this dissertation, we primarily explore this strategy due to its efficiency.

Generalizability and Speed To address both the generalizability and speed challenges, we propose a novel method, One-2-3-45 (introduced in Chapter 2), for generating 3D textured meshes of any object from a single image in a feed-forward manner. Instead of following the

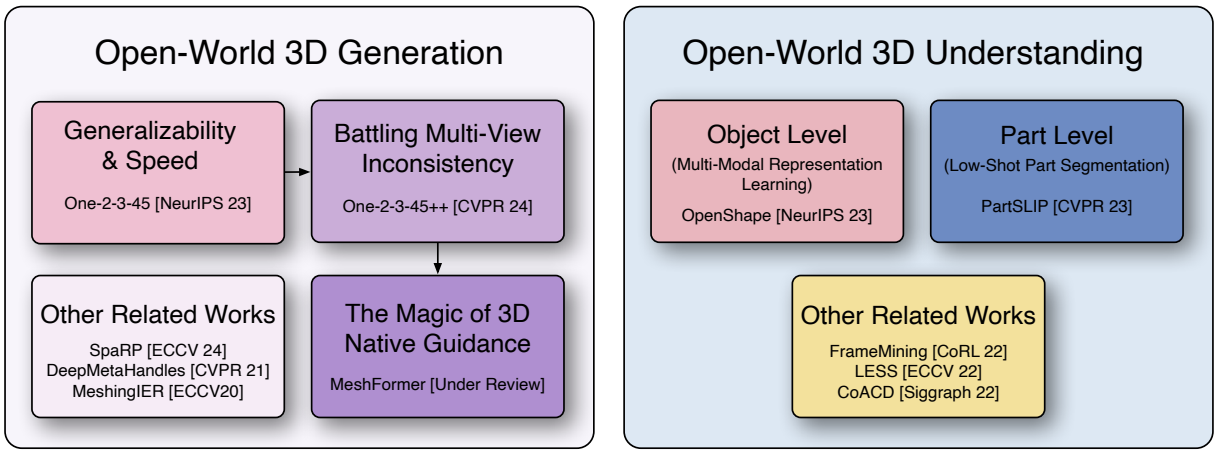


Figure 1.1. Overview of contributions. We primarily focus on 3D object generation and 3D object understanding in an open-world context. For each, we explore several strategies to effectively utilize various priors and identify a series of crucial problems, offering proposed solutions. Our efforts have significantly enhanced the generalizability of many previous 3D understanding and generation tasks.

optimization-based paradigm, One-2-3-45 propose a novel approach to utilize 2D prior models for 3D generation. At the core of the method is the combination of a 2D diffusion model with a cost-volume-based 3D reconstruction technique. Specifically, we leverage a recent 2D diffusion model, Zero123 [170], which is fine-tuned on Stable Diffusion to predict novel views of the input image given camera transformations. We use it to generate multi-view predictions of the input single image and then feed the predicted multi-view images into a cost-volume-based feed-forward network for 3D reconstruction. Without costly optimizations, One-2-3-45 reconstructs open-world 3D shapes for the first time in just 45 seconds.

Battling Multi-View Inconsistency While One-2-3-45 presents the first efficient feed-forward pipeline for open-world 3D generation, the multi-view images predicted by 2D diffusion models are not always perfect or 3D-consistent, which compromises the 3D reconstruction results. To overcome these shortcomings and deliver significantly enhanced robustness and quality, we introduce One-2-3-45++ (introduced in Chapter 3), which primarily focuses on improving the 3D consistency of the generated multi-view images and the robustness of the reconstruction model when faced with inconsistent multi-view images. Taking a single image of any object as input,

One-2-3-45++ includes two primary stages: 2D multi-view generation and 3D reconstruction. In the initial phase, rather than predicting each view independently as in Zero123, One-2-3-45++ predicts consistent multi-view images jointly, improving 3D consistency among the multi-view images. In the second stage, One-2-3-45++ employs a multi-view conditioned 3D-diffusion-based module to predict the textured mesh in a coarse-to-fine fashion. The multi-view conditional images act as a blueprint for 3D reconstruction, facilitating a zero-shot hallucination capability. Concurrently, the 3D diffusion network excels in lifting the inconsistent multi-view images, thanks to its ability to harness a broad spectrum of priors extracted from the 3D dataset. One-2-3-45++ efficiently generates 3D meshes with realistic textures in under a minute.

The Magic of 3D Native Guidance Following our proposed paradigm of combining 2D diffusion models for multi-view prediction with feed-forward models for sparse-view 3D reconstruction, many recent works have explored how to improve the consistency of multi-view predictions. Regarding the feed-forward model that converts multi-view images into 3D, a family of works, pioneered by the large reconstruction model (LRM) [97], stands out due to its impressive performance. It combines large-scale transformer models with the triplane representation and trains the model primarily using a rendering loss. Although straightforward, these methods typically require over a hundred GPUs to train. Moreover, due to their reliance on volume rendering, these methods have difficulty extracting high-quality meshes.

To address these limitations, we present MeshFormer (introduced in Chapter 4), an open-world sparse-view mesh reconstruction model. Instead of representing 3D data as "2D planes" and training a "black box" transformer model optimized only for rendering loss, we find that by incorporating various types of 3D-native priors into the model design—such as 3D representations, network architecture, supervision signals, and input guidance—our model can significantly improve both mesh quality and training efficiency. Specifically, instead of using a triplane representation, we store features in 3D sparse voxels and combine transformers with 3D convolutions to leverage an explicit 3D structure and projective bias. In addition to sparse-view RGB input, we require the network to take input and generate corresponding normal maps. The

Figure 1.2. Qualitative examples of MeshFormer. Given a sparse set (e.g., six) of multi-view RGB images and their corresponding normal maps as input, MeshFormer reconstructs high-quality 3D textured meshes with fine-grained, sharp geometric details in a feed-forward pass that takes just a few seconds.

input normal maps can be predicted by 2D diffusion models, significantly aiding in the guidance and refinement of the geometry learning process. Moreover, by combining Signed Distance Function (SDF) supervision with surface rendering, we directly learn to generate high-quality meshes without the need for complex multi-stage training processes. By incorporating these explicit 3D biases, MeshFormer can be trained efficiently and deliver high-quality textured meshes with fine-grained geometric details, as shown in Figure 1.2.

Other Related Projects The above three works focus on single-image-to-3D. We also propose an extension project, SpaRP (introduced in Section 3.5), which takes unposed sparse-view images

as input. By following a similar strategy to exploit 2D diffusion models, SpaRP reconstructs a 3D textured mesh from sparse-view images in a feed-forward fashion and estimates the relative camera poses for the input sparse-view images.

In addition to MeshFormer, we introduce two additional projects (introduced in Section 4.5) that leverage 3D-native priors to reduce the need for 3D ground truth in 3D generative models. One of the projects learns to generate all plausible deformations of a 3D mesh, which can be used for the self-proliferation of 3D shapes. The other project explores a learning-based point cloud meshing algorithm by explicitly learning to add triangles.

1.5.2 Open-World 3D Understanding

Unlike 3D object generation, which requires the creation of novel 3D shapes, 3D understanding focuses on analyzing various properties of input 3D objects. This includes low-level aspects like appearance and geometry, as well as high-level features such as structure, semantics, mobility, functionality, and affordance. To achieve open-world capabilities in 3D understanding, one can leverage extensive knowledge from 2D models, often by using 2D images as intermediaries and projecting 3D data into 2D space. Two primary strategies for utilizing 2D pre-trained models in this context are inference-only multi-view fusion and knowledge distillation.

The first strategy, inference-only multi-view fusion, involves using 2D pre-trained models exclusively during the inference stage. In this strategy, 3D objects are projected onto 2D space, and the resulting multi-view images are processed by 2D models to generate desired outputs such as labels or segmentation masks. These outputs are then unprojected and fused back into 3D. While this approach is relatively straightforward and typically requires minimal 3D training, it faces challenges such as potential 3D inconsistency when fusing multi-view 2D results into 3D. Additionally, it may result in longer runtimes for processing each 3D shape.

The second strategy involves training a 3D-native network while distilling knowledge or priors from 2D diffusion models, using 2D images as a bridge. This can be achieved through explicit latent feature alignment between 3D-native models and 2D pre-trained models or by using

the 2D models to generate pseudo-labels or guidance during the 3D training process. Although this strategy typically requires extensive 3D training and relies on large-scale 3D datasets, it offers faster inference speeds, and successful training often produces more 3D-consistent and robust results compared to the inference-only multi-view fusion method.

In this dissertation, we explore these two strategies across two different tasks: one focusing on learning open-world multi-modal 3D representations, and the other concentrating on a more localized understanding, specifically in the context of open-world 3D part segmentation.

Multi-Modal 3D Representation Learning The OpenShape project (discussed in Chapter 5) introduces a method for learning multi-modal joint representations across text, images, and 3D point clouds, with the aim of extending the success of 2D CLIP [234] to the 3D domain. We specifically train a 3D-native point cloud encoder to extract 3D shape representations and align them with CLIP text and image embeddings using a multi-modal contrastive learning framework. The extent of knowledge that can be distilled, however, is contingent upon the size and quality of the training triplets (3D-2D-text). To scale up 3D representations for open-world 3D shape understanding, we focus on expanding the training dataset by ensembling multiple 3D datasets and introducing strategies to automatically filter and enrich noisy text descriptions. We also explore methods for scaling 3D backbone networks and introduce a novel hard negative mining module to enhance training efficiency.

We evaluate OpenShape using zero-shot 3D classification benchmarks, demonstrating its superior capabilities in open-world recognition. Notably, OpenShape is the first model capable of classifying over a thousand object categories with excellent zero-shot accuracy (46.8% compared to less than 10% for existing methods). Furthermore, we demonstrate that our learned embeddings capture a wide range of visual and semantic concepts (e.g., subcategories, color, shape, style) and enable fine-grained text-3D and image-3D interactions. Due to their alignment with CLIP embeddings, these learned shape representations can also be integrated with off-the-shelf CLIP-based models for various applications, such as point cloud captioning and point cloud-conditioned image generation.

Low-Shot Part Segmentation While OpenShape learns powerful 3D representations for open-world 3D shapes, it primarily captures global attributes of the input shape but may fail to discern its part structure. However, open-world 3D part segmentation is crucial for enabling a wide range of applications, such as robotic manipulation, AR/VR, and shape analysis and synthesis. In Chapter 6, we explore a novel approach for low-shot part segmentation of 3D point clouds by leveraging a pretrained image-language model, GLIP [377], which achieves superior performance on open-vocabulary 2D detection. We transfer the rich knowledge from 2D to 3D through GLIP-based part detection on point cloud renderings and a novel 2D-to-3D label lifting algorithm. We also utilize multi-view 3D priors and few-shot prompt tuning to significantly boost performance.

Extensive evaluations show that our method enables excellent zero-shot 3D part segmentation. Our few-shot version not only outperforms existing few-shot approaches by a large margin but also achieves highly competitive results compared to fully supervised counterparts. Furthermore, we demonstrate that our method can be directly applied to iPhone-scanned point clouds without significant domain gaps.

Other Related Projects: Regarding 3D representation learning, we found that when the input consists of multiple 3D objects interacting with each other, the choice of input point cloud coordinate frames significantly impacts learning from 3D point clouds (see Section 5.5).

Additionally, I have two other projects related to 3D part segmentation (see Section 6.5). One focuses on segmenting LiDAR point clouds for autonomous driving, while the other aims to decompose 3D shapes into sets of approximately convex components. Both projects aim to minimize reliance on 3D training labels, exploring either label-efficient or label-free approaches. These two projects, alongside the open-world 3D part segmentation work, underscore the critical importance of part-level structure understanding of 3D data in various scenarios.

Chapter 2

Open-World 3D Generation: Generalizability and Speed

Single image 3D reconstruction, the task of reconstructing a 3D model of an object from a single 2D image, is a long-standing problem in the computer vision community and is crucial for a wide range of applications, such as robotic object manipulation and navigation, 3D content creation, as well as AR/VR [40, 195, 355]. The problem is challenging as it requires not only the reconstruction of visible parts but also the hallucination of invisible regions. Consequently, this problem is often ill-posed and corresponds to multiple plausible solutions because of insufficient evidence from a single image. On the other hand, humans can adeptly infer unseen 3D content based on our extensive knowledge of the 3D world. To endow intelligent agents with this ability, many existing methods [41, 43, 62, 75, 108, 123, 160, 163, 322, 336, 351] exploit class-specific priors by training 3D generative networks on 3D shape datasets [22]. However, these methods often fail to generalize to unseen categories, and their reconstruction quality is constrained by the limited size of public 3D datasets.

In this work, we pursue a generic solution to turn an image of any object, regardless of its category, into a high-quality 3D textured mesh. To achieve this, we propose a novel approach that can effectively utilize the strong priors learned by 2D diffusion models for 3D reconstruction. Compared to 3D data, 2D images are more readily available and scalable. Recent 2D generative models (e.g., DALL-E [236, 237], Imagen [249], and Stable Diffusion [247])

Figure 2.1. One-2-3-45 reconstructs a full 360° mesh of any object in 45 seconds given a single image of it. In each example, we showcase the input image in the left column, alongside the generated textured and textureless meshes from three different views.

and visual-language models (e.g., CLIP [234]) have made significant strides by pre-training on Internet-scale image datasets. Since they learn a wide range of visual concepts and possess strong priors about our 3D world, it is natural to marry 3D tasks with them. Consequently, an emerging body of research [96, 110, 154, 201, 225], as exemplified by DreamField [110], DreamFusion [225], and Magic3D [154], employs 2D diffusion models or vision language models to assist 3D generative tasks. The common paradigm of them is to perform per-shape optimization with differentiable rendering and the guidance of the CLIP model or 2D diffusion models. While many other 3D representations have been explored, neural fields are the most commonly used representation during optimization.

Although these optimization-based methods have achieved impressive results on both

text-to-3D [110, 154, 225] and image-to-3D tasks [196, 257], they face some common dilemmas: (a) **time-consuming**. Per-shape optimization typically involves tens of thousands of iterations of full-image volume rendering and prior model inferences, resulting in typically tens of minutes per shape. (b) **memory intensive**. Since the full image is required for the 2D prior model, the volume rendering can be memory-intensive when the image resolution goes up. (c) **3D inconsistent**. Since the 2D prior model only sees a single view at each iteration and tries to make every view look like the input, they often generate 3D inconsistent shapes (e.g., with two faces, or the Janus problem [196, 225]). (d) **poor geometry**. Many methods utilize the density field as the representation in volume rendering. It is common that they produce good RGB renderings but extracting high-quality mesh tends to be difficult.

In this chapter, instead of following the common optimization-based paradigm, we propose a novel approach to utilize 2D prior models for 3D modeling. At the heart of our approach is the combination of a 2D diffusion model with a cost-volume-based 3D reconstruction technique, enabling the reconstruction of a high-quality 360° textured mesh from a single image in a feed-forward pass without per-scene optimization. Specifically, we leverage a recent 2D diffusion model, Zero123 [170], which is fine-tuned on Stable Diffusion [247] to predict novel views of the input image given the camera transformation. We utilize it to generate multi-view predictions of the input single image so that we can leverage multi-view 3D reconstruction techniques to obtain a 3D mesh. There are two challenges associated with reconstruction from synthesized multi-view predictions: (a) the inherent lack of perfect consistency within the multi-view predictions, which can lead to severe failures in optimization-based methods such as NeRF methods [25, 202]. (b) the camera pose of the input image is required but unknown. To tackle them, we build our reconstruction module upon a cost volume-based neural surface reconstruction approach, SparseNeuS [184], which is a variant of MVSNeRF [26]. Additionally, we introduce a series of essential training strategies that enable the reconstruction of 360-degree meshes from inherently inconsistent multi-view predictions. We also propose an elevation estimation module that estimates the elevation of the input shape in Zero123's canonical coordinate system, which

is used to compute the camera poses required by the reconstruction module.

By integrating the three modules of multi-view synthesis, elevation estimation, and 3D reconstruction, our method can reconstruct 3D meshes of any object from a single image in a feed-forward manner. Without costly optimizations, our method reconstructs 3D shapes in significantly less time, e.g., in just 45 seconds. Our method favors better geometry due to the use of SDF representations, and generates more consistent 3D meshes, thanks to the camera-conditioned multi-view predictions. Moreover, our reconstruction adheres more closely to the input image compared to existing methods. See Figure 2.1 for some of our example results. We evaluate our method on both synthetic data and real images and demonstrate that our method outperforms existing methods in terms of both quality and efficiency.

2.1 Related Work

2.1.1 3D Generation Guided by 2D Prior Models

Recently, 2D generative models (e.g., DALL-E [236, 237], Imagen [249], and Stable Diffusion [247]) and vision-language models (e.g., CLIP [234]) have learned a wide range of visual concepts by pre-training on Internet-scale image datasets. They possess powerful priors about our 3D world and have inspired a growing body of research to employ 2D prior models for assisting 3D understanding [162, 168] and generative tasks. Exemplified by DreamField [110], DreamFusion [225], and Magic3D [154], a line of works follows the paradigm of per-shape optimization. They typically optimize a 3D representation (i.e., NeRF, mesh, SMPL human model) and utilize differentiable rendering to generate 2D images from various views. The images are then fed to the CLIP model [9, 20, 96, 110, 114, 129, 139, 179, 201, 342] or 2D diffusion model [51, 154, 196, 200, 225, 235, 257, 279, 297, 340, 395] for calculating the loss functions, which are used to guide the 3D shape optimization. In addition to optimization-based 3D shape generation, some works train a 3D generative model but leverage the embedding space of CLIP [38, 180, 252], and some works focus on generating textures or materials for input

meshes using 2D models' prior [27, 200, 201, 245, 317].

2.1.2 Single Image to 3D

Before the emergence of CLIP and large-scale 2D diffusion models, people often learn 3D priors from 3D synthetic data [22] or real scans [241]. Unlike 2D images, 3D data can be represented in various formats and numerous representation-specific 3D generative models have been proposed. By combining 2D image encoder and 3D generators, they generate 3D data in various representations, including 3D voxels [43, 75, 329, 335, 336, 351], point clouds [4, 62, 78, 197, 362, 373], polygon meshes [123, 213, 300, 322], and parametric models [223, 399, 400]. Recently, there has been an increasing number of work on learning to generate a 3D implicit field from a single image [67, 84, 108, 112, 199, 204, 211, 222, 250, 327, 345].

As previously mentioned, several recent works leverage 2D diffusion models to perform per-shape optimization, allowing for the text-to-3D task [110, 154, 225] given that diffusion models are typically conditioned on text. To enable the generation of 3D models from a single image, some works [51, 196, 200] utilize textual inversion [66], to find the best-matching text embedding for the input image, which is then fed into a diffusion model. NeuralLift-360 [101] adds a CLIP loss to enforce similarity between the rendered image and the input image. 3DFuse [257] finetunes the Stable Diffusion model with LoRA layers [101] and a sparse depth injector to ensure greater 3D consistency. A recent work Zero123 [170, 262] finetunes the Stable Diffusion model [249] to generate a novel view of the input image based on relative camera pose. In addition to these methods, OpenAI trains a 3D native diffusion model Point-E [215], which uses several million internal 3D models to generate point clouds. Very recently, they published another model Shap-E [121] which is trained to generate parameters of implicit functions that can be used for producing textured meshes or neural radiance fields.

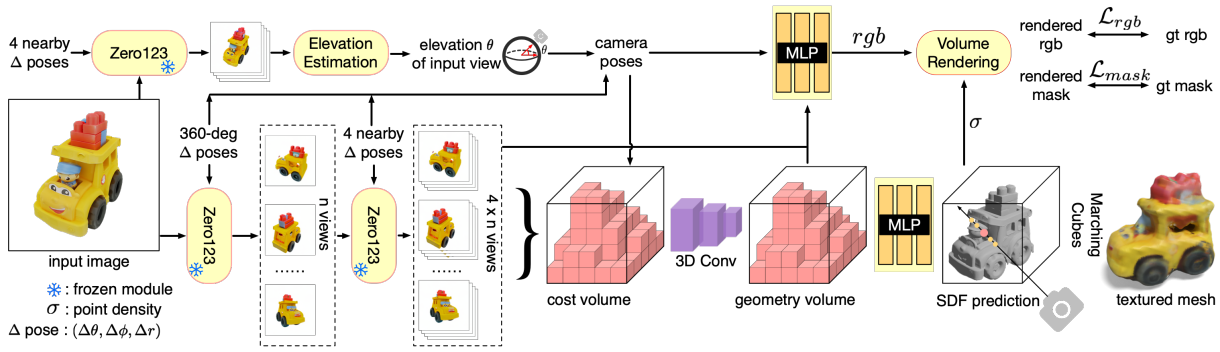


Figure 2.2. Our method consists of three primary components: (a) **Multi-view synthesis**: we use a view-conditioned 2D diffusion model, Zero123 [170], to generate multi-view images in a two-stage manner. The input of Zero123 includes a single image and a relative camera transformation, which is parameterized by the relative spherical coordinates $(\Delta\theta, \Delta\phi, \Delta r)$. (b) **Pose estimation**: we estimate the elevation angle θ of the input image based on four nearby views generated by Zero123. We then obtain the poses of the multi-view images by combining the specified relative poses with the estimated pose of the input view. (c) **3D reconstruction**: We feed the multi-view posed images to an SDF-based generalizable neural surface reconstruction module for 360° mesh reconstruction.

2.1.3 Generalizable Neural Reconstruction

Traditional NeRF-like methods [202, 302] use a neural network to represent a single scene and require per-scene optimization. However, some approaches aim to learn priors across scenes and generalize to novel scenes. These methods typically take a few source views as input and leverage 2D networks for extracting 2D features. The pixel features are then unprojected into 3D space, and a NeRF-based rendering pipeline is applied on top of them. In this way, they can generate a 3D implicit field given a few source views in a single feed-forward pass. Among the methods, some [92, 135, 176, 241, 287, 291, 305, 356, 367] directly aggregate 2D features with MLPs or transformers, while others explicitly construct the 3D feature/cost volume [26, 119, 184, 384], and utilize the voxel feature for decoding density and color. In addition to the density field representation, some methods such as SparseNeuS [184] and VolRecon [243] utilize SDF representations for geometry reconstruction.

2.2 Proposed Method: One-2-3-45

Our overall pipeline is illustrated in Figure 2.2. In Section 2.2.1, we introduce a view-conditioned 2D diffusion model, Zero123 [170], which is used to generate multi-view images. In Section 2.2.2, we show that traditional NeRF-based and SDF-based methods fail to reconstruct high-quality meshes from inconsistent multi-view predictions even given ground truth camera poses. Therefore, in Section 2.2.3, we propose a cost volume-based neural surface reconstruction module that can be trained to handle inconsistent multi-view predictions and reconstruct a 3D mesh in a single feed-forward pass. Specifically, we build upon the SparseNeuS [184] and introduce several critical training strategies to support 360° mesh reconstruction. Additionally, in Section 2.2.4, we demonstrate the necessity of estimating the pose of the input view in Zero123's canonical space for 3D reconstruction. While the azimuth and radius can be arbitrarily specified, we propose a novel module that utilizes four nearby views generated by Zero123 to estimate the elevation of the input view.

2.2.1 Zero123: View-Conditioned 2D Diffusion

Recent 2D diffusion models [237, 247, 249] have demonstrated the ability to learn a wide range of visual concepts and strong priors by training on internet-scale data. While the original diffusion models mainly focused on the task of text-to-image, recent work [101, 381] has shown that fine-tuning pretrained models allows us to add various conditional controls to the diffusion models and generate images based on specific conditions. Several conditions, such as canny edges, user scribbles, depth, and normal maps, have already proven effective [381].

The recent work Zero123 [170] shares a similar spirit and aims to add viewpoint condition control for the Stable Diffusion model [247]. Specifically, given a single RGB image of an object and a relative camera transformation, Zero123 aims to control the diffusion model to synthesize a new image under this transformed camera view. To achieve this, Zero123 fine-tunes the Stable Diffusion on paired images with their relative camera transformations, synthesized from a

large-scale 3D dataset [50]. During the creation of the fine-tuning dataset, Zero123 assumes that the object is centered at the origin of the coordinate system and uses a spherical camera, i.e., the camera is placed on the sphere's surface and always looks at the origin. For two camera poses (θ_1, ϕ_1, r_1) and (θ_2, ϕ_2, r_2), where θ_i, ϕ_i, and r_i denote the polar angle, azimuth angle, and radius, their relative camera transformation is parameterized as $(\theta_2 - \theta_1, \phi_2 - \phi_1, r_2 - r_1)$. They aim to learn a model f, such that $f(x_1, \theta_2 - \theta_1, \phi_2 - \phi_1, r_2 - r_1)$ is perceptually similar to x_2, where x_1 and x_2 are two images of an object captured from different views. Zero123 finds that such fine-tuning enables the Stable Diffusion model to learn a generic mechanism for controlling the camera viewpoints, which extrapolates outside of the objects seen in the fine-tuning dataset.

2.2.2 Can NeRF Optimization Lift Multi-View Predictions to 3D?

Given a single image of an object, we can utilize Zero123 [170] to generate multi-view images, but can we use traditional NeRF-based or SDF-based methods [25, 302] to reconstruct high-quality 3D meshes from these predictions? We conduct a small experiment to test this hypothesis. Given a single image, we first generate 32 multi-view images using Zero123, with camera poses uniformly sampled from the sphere surface. We then feed the predictions to a NeRF-based method (TensoRF [202]) and an SDF-based method (NeuS [302]), which optimize density and SDF fields, respectively. However, as shown in Figure 2.3, both methods fail to produce satisfactory results, generating numerous distortions and floaters. This is primarily due to the inconsistency of Zero123's predictions. In Figure 2.4, we compare Zero123's predictions with ground-truth renderings. We can see that the overall PSNR is not very high, particularly when the input relative pose is large or the target pose is at unusual locations (e.g., from the bottom or the top). However, the mask IoU (most regions are greater than 0.95) and CLIP similarity are relatively good. This suggests that Zero123 tends to generate predictions that are perceptually similar to the ground truth and have similar contours or boundaries, but the pixel-level appearance may not be exactly the same. Nevertheless, such inconsistencies between the source views are already fatal to traditional optimization-based methods. Although the

Figure 2.3. NeRF-based method [202] and SDF-based method [302] fail to reconstruct high-quality meshes given multi-view images predicted by Zero123. See Figure 2.1 for our reconstruction results.

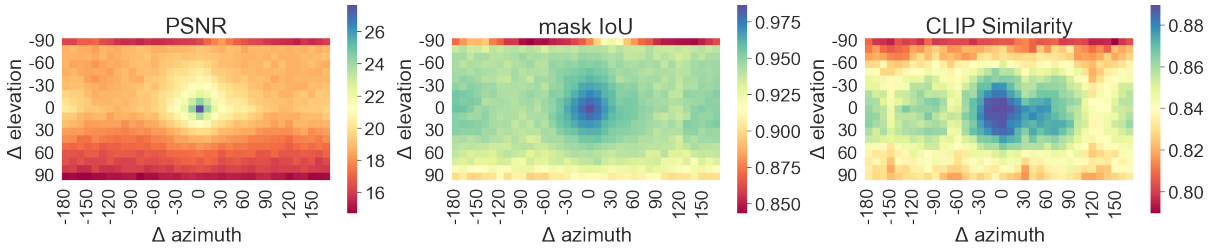


Figure 2.4. We analyze the prediction quality of Zero123 by comparing its predictions to ground truth renderings across various view transformations. For each view transformation, we report the average PSNR, mask IoU, and CLIP similarity of 100 shapes from the Objaverse [50] dataset. The prediction mask is calculated by considering foreground objects (i.e., non-white regions). Zero123 provides more accurate predictions when the view transformation is small.

original Zero123 paper proposes another method for lifting its multi-view predictions, we will demonstrate in experiments that it also fails to yield perfect results and entails time-consuming optimization.

2.2.3 Neural Surface Reconstruction from Imperfect Multi-View Predictions

Instead of using optimization-based approaches, we base our reconstruction module on a generalizable SDF reconstruction method SparseNeuS [184], which is essentially a variant of the MVSNeRF [26] pipeline that combines multi-view stereo, neural scene representation, and volume rendering. As illustrated in Figure 2.2, our reconstruction module takes multiple source images with corresponding camera poses as input and generates a textured mesh in a single feed-forward pass. In this section, we will first briefly describe the network pipeline of the module and then explain how we train the module, select the source images, and generate textured meshes. Additionally, in Section 2.2.4, we will discuss how we generate the camera

poses for the source images.

As shown in Figure 2.2, our reconstruction module takes m posed source images as input. The module begins by extracting m 2D feature maps using a 2D feature network. Next, the module builds a 3D cost volume whose contents are computed by first projecting each 3D voxel to m 2D feature planes and then fetching the variance of the features across the m projected 2D locations. The cost volume is then processed using a sparse 3D CNN to obtain a geometry volume that encodes the underlying geometry of the input shape. To predict the SDF at an arbitrary 3D point, an MLP network takes the 3D coordinate and its corresponding interpolated features from the geometry encoding volume as input. To predict the color of a 3D point, another MLP network takes as input the 2D features at the projected locations, interpolated features from the geometry volume, and the viewing direction of the query ray relative to the viewing direction of the source images. The network predicts the blending weights for each source view, and the color of the 3D point is predicted as the weighted sum of its projected colors. Finally, an SDF-based rendering technique is applied on top of the two MLP networks for RGB and mask rendering [302]. In each iteration, we randomly choose one view to build the cost volume and another view for rendering supervision.

2-Stage Source View Selection and Groundtruth-Prediction Mixed Training. Although the original SparseNeuS [184] paper only demonstrated frontal view reconstruction, we have extended it to reconstruct 360-degree meshes in a single feed-forward pass by selecting source views in a particular way. Specifically, our reconstruction model is trained on a 3D object dataset while freezing Zero123. We follow Zero123 to normalize the training shapes and use a spherical camera model. For each shape, we first render n ground-truth RGB images from n camera poses uniformly placed on the sphere. For each of the n views, we use Zero123 to predict four nearby views. During training, we feed all $4 \times n$ predictions with ground-truth poses into the reconstruction module and randomly choose one of the n ground-truth RGB images views as the target view. We call this view selection strategy as *2-stage source view selection*. We supervise the training with both the ground-truth RGB and mask values. In this way, the module can learn

to handle the inconsistent predictions from Zero123 and reconstruct a consistent 360° mesh. We argue that our two-stage source view selection strategy is critical since uniformly choosing $n \times 4$ source views from the sphere surface would result in larger distances between the camera poses. However, cost volume-based methods [26, 119, 184] typically rely on very close source views to find local correspondences. Furthermore, as shown in Figure 2.4, when the relative pose is small (e.g., 10 degrees apart), Zero123 can provide very accurate and consistent predictions and thus can be used to find local correspondences and infer the geometry.

During training, we utilize n ground-truth renderings in the initial stage. We find that employing n predicted images at this stage would suffer from notable inconsistencies across different views, complicating the network's ability to learn sharp details (see examples in ablation study). However, during inference, we can replace the n ground-truth renderings with Zero123 predictions, as shown in Figure 2.2, the network can automatically generalize to some extent. We will show in the experiments that this groundtruth-prediction mixed training strategy is also important. To export the textured mesh, we use marching cubes [185] to extract the mesh from the predicted SDF field and query the color of the mesh vertices as described in [302]. Although our reconstruction module is trained on a 3D dataset, we find that it mainly relies on local correspondences and can generalize to unseen shapes very well.

2.2.4 Camera Pose Estimation

Our reconstruction module requires camera poses for the $4 \times n$ source view images. Note that we adopt Zero123 for image synthesis, which parameterizes cameras in a canonical spherical coordinate frame, (θ, ϕ, r), where θ, ϕ and r represent the elevation, azimuth, and radius. While we can arbitrarily adjust the azimuth angle ϕ and the radius r of all source view images simultaneously, resulting in the rotation and scaling of the reconstructed object accordingly, this parameterization requires knowing the absolute elevation angle θ of one camera to determine the relative poses of all cameras in a standard XYZ frame. More specifically, the relative poses between camera (θ_0, ϕ_0, r_0) and camera $(\theta_0 + \Delta\theta, \phi_0 + \Delta\phi, r_0)$ vary for different

θ_0 even when $\Delta\theta$ and $\Delta\phi$ are the same. Because of this, changing the elevation angles of all source images together (e.g., by 30 degrees up or 30 degrees down) will lead to the distortion of the reconstructed shape (see Figure 2.11 for examples).

Therefore, we propose an elevation estimation module to infer the elevation angle of the input image. First, we use Zero123 to predict four nearby views of the input image. Then we enumerate all possible elevation angles in a coarse-to-fine manner. For each elevation candidate angle, we compute the corresponding camera poses for the four images and calculate a reprojection error for this set of camera poses to measure the consistency between the images and the camera poses. The elevation angle with the smallest reprojection error is used to generate the camera poses for all $4 \times n$ source views by combining the pose of the input view and the relative poses.

2.3 Experiments

2.3.1 Implementation Details

For each input image, we generate $n = 8$ images by choosing camera poses uniformly placed on the sphere surface and then generate 4 local images (10° apart) for each of the 8 views, resulting in 32 source-view images for reconstruction. During training, we freeze the Zero123 [170] model and train our reconstruction module on the Objaverse-LVIS [50] dataset, which contains 46K 3D models in 1,156 categories. We use BlenderProc [55] to render ground-truth RGB images. For images with background, we utilize an off-the-shelf segmentation network SAM [131] with bounding-box prompts for background removal.

2.3.2 Single Image to 3D Mesh

We present qualitative examples of our method in Figures 2.1 and 2.5, illustrating its effectiveness in handling both synthetic images and real images. We also compare One-2-3-45 with existing zero-shot single image 3D reconstruction approaches, including Point-E [215],

Figure 2.5. Qualitative examples of One-2-3-45 for both synthetic and real images. Each triplet showcases an input image, a textured mesh, and a textureless mesh.

Table 2.1. Quantitative Comparison on GSO [59] and Objaverse [50] datasets.

	Prior Source	F-Score			CLIP Similarity			Time
		GSO	Obj.	avg.	GSO	Obj.	avg.	
Point-E [215]	internal 3D data	81.0	81.0	81.0	74.3	78.5	76.4	78s
Shap-E [121]		83.4	81.2	82.3	**79.6**	**82.1**	**80.9**	27s
Zero123+SD [170]	2D diffusion models	75.1	69.9	72.5	71.0	72.7	71.9	∼15min
RealFusion [196]		66.7	59.3	63.0	69.3	69.5	69.4	∼90min
3DFuse [257]		60.7	60.2	60.4	71.4	74.0	72.7	∼30min
Ours		**84.0**	**83.1**	**83.5**	76.4	79.7	78.1	45s

Shap-E [121], Zero123 (Stable Dreamfusion version) [170], 3DFuse [257], and RealFusion [196]. Among them, Point-E and Shap-E are two 3D native diffusion models released by OpenAI, which are trained on several million internal 3D data, while others are optimization-based approaches leveraging priors from Stable Diffusion [247].

Figure 2.6 presents the qualitative comparison. While most methods can generate plausible 3D meshes from a single image, notable differences exist among them in terms of geometry quality, adherence to the input, and overall 3D consistency. In terms of geometry quality, approaches like RealFusion [196] and 3DFuse [257], which optimize a neural radiance field, face challenges in extracting high-quality meshes. Likewise, Point-E [215] produces a

Figure 2.6. We compare One-2-3-45 with Point-E [215], Shap-E [121], Zero123 (Stable Dreamfusion version) [170], 3DFuse [257], and RealFusion [196]. In each example, we present both the textured and textureless meshes. As 3DFuse [257] and RealFusion [196] do not natively support the export of textured meshes, we showcase the results of volume rendering instead.

sparse point cloud as its output, resulting in numerous holes on the reconstructed meshes. In contrast, our approach utilizes an SDF presentation and favors better geometry. Regarding adherence to the input, we observe that most baseline methods struggle to preserve the similarity to the input image. Although Shap-E performs slightly better, it still produces lots of failure cases (see the backpack without shoulder straps, distorted shoe, and stool with three legs). In contrast, our approach leverages a powerful 2D diffusion model to directly produce high-quality multi-view images, rather than relying on 3D space hallucination. This strategy provides better adherence to the input views, alleviates the burden of the 3D reconstruction module, and yields results that are more finely attuned to the input. Furthermore, many approaches encounter challenges in achieving consistent 3D results (also known as the Janus problem [196, 225]), as

Figure 2.7. Unlike previous approaches, One-2-3-45 doesn't suffer from the multi-face problem (Janus problem).

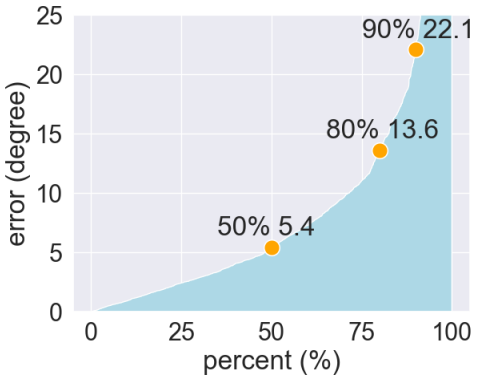


Figure 2.8. Error distribution of predicted elevations. The median and average are 5.4 and 9.7 degrees.

highlighted in Figure 2.7 (two-handle mug, multi-face Mario, and two-face backpack). One of the contributing factors to this issue is that several methods optimize each view independently, striving to make each view resemble the input. In contrast, our method capitalizes on the view-conditioned 2D diffusion model, inherently enhancing 3D consistency.

We also quantitatively compare the approaches on Objaverse [50] and GoogleScannedObjects (GSO) [59] datasets. For each dataset, we randomly choose 20 shapes and render a single image per shape for evaluation. To align the predictions with the ground-truth mesh, we linearly search the scaling factor and the rotation angle, apply Iterative Closest Point (ICP) for sampled point clouds, and select the one with the most number of inliers. We follow RealFusion [196] to report F-score (with a threshold of 0.05) and CLIP similarity, and the runtime on an A100 GPU.

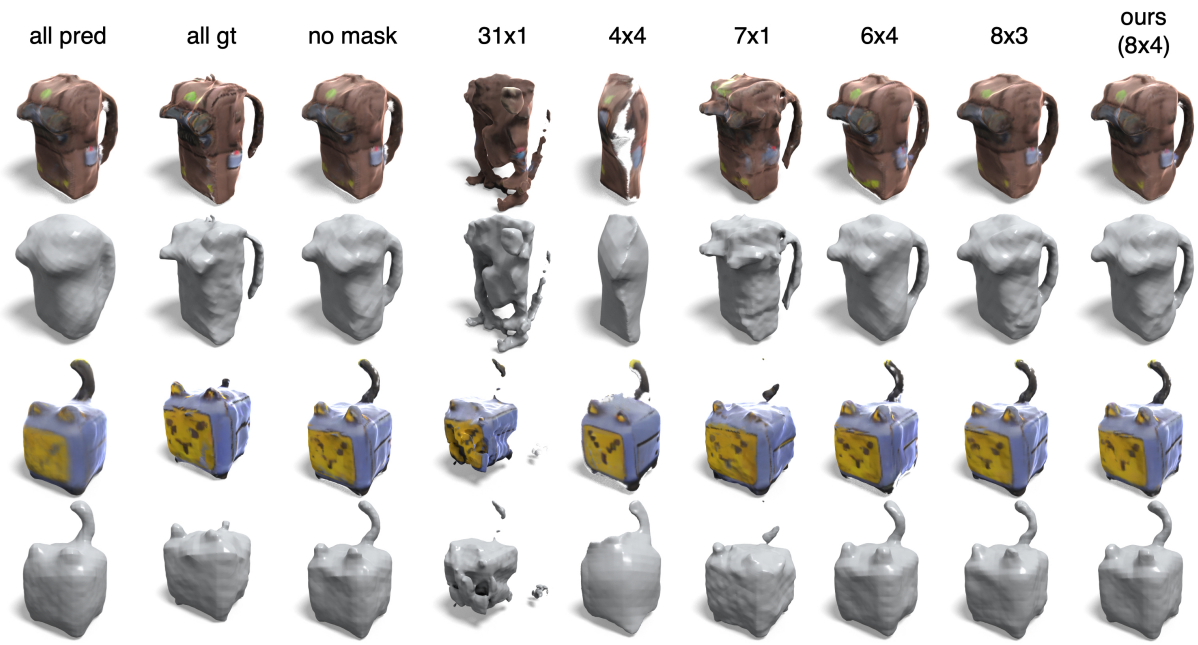


Figure 2.9. Ablations on training strategies of the reconstruction module and the number of views.

As shown in Table 2.1, our method outperforms all baseline approaches in terms of F-Score. As for CLIP similarity, we surpass all methods except a concurrent work Shap-E [121]. We find that CLIP similarity is very sensitive to the color distribution and less discriminative in local geometry variations (i.e., the number of legs of a stool, the number of handles of a mug). Regarding running time, our method demonstrates a notable advantage over optimization-based approaches and performs on par with 3D native diffusion models, such as Point-E [215] and Shap-E [121]. Specifically, our 3D reconstruction module reconstructs a 3D mesh in approximately 5 seconds, with the remaining time primarily spent on Zero123 predictions, which takes roughly 1 second per image on an A100 GPU.

2.3.3 Ablation Study

Training strategies. We ablate our training strategies in Figure 2.9. We found that without our 2-stage source view selection strategy, a network trained to consume 31 uniformly posed Zero123 predictions (fourth column) suffers from severe inconsistency among source views,

causing the reconstruction module to fail completely. If we feed only 7 source views (sixth column) without the four nearby views, the reconstruction fails to capture local correspondence and cannot reconstruct fine-grained geometry. During training, we first render n ground-truth renderings and then use Zero123 to predict four nearby views for each of them. If we train directly on 8×4 ground-truth renderings without Zero123 prediction during training (second column), it fails to generalize well to Zero123 predictions during inference, with many missing regions. Instead, if we replace the n ground-truth renderings with n Zero123 predictions during training (first column), the network fail to generate sharp details (see the strips of the backpack).

Elevation estimation. Our reconstruction module relies on accurate elevation angles of the input view. In Figure 2.11, we demonstrate the impact of providing incorrect elevation angles (e.g., altering the elevation angles of source views by $\pm 30^\circ$), which results in distorted reconstruction results. Instead, utilizing our predicted elevation angles can perfectly match results with ground truth elevations. We also quantitatively test our elevation estimation module by rendering 1,700 images from random camera poses. As shown in Figure 2.8, our elevation estimation module predicts accurate elevations.

Number of source views. In Figure 2.9, we also investigate the impact of varying the number of source views on 3D reconstruction. We observe that our method is not very sensitive to the number of views as long as the reconstruction module is retrained with the corresponding setting.

360° **reconstruction vs. multi-view fusion.** While our method reconstructs a 360° mesh in a single pass, most existing generalizable neural reconstruction approaches [26, 119, 184] primarily focus on frontal view reconstruction. An alternative approach is to independently infer the geometry for each view and subsequently fuse them together. However, we have observed that this strategy often struggles with multi-view fusion due to inconsistent Zero123 predictions, as illustrated in Figure 2.10.

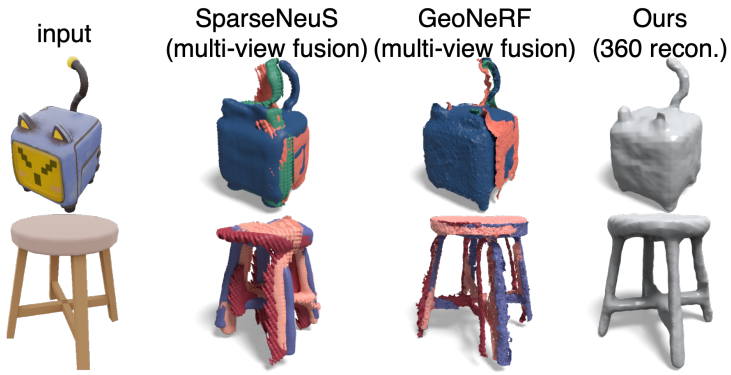


Figure 2.10. 360° reconstruction vs. multi-view fusion. Meshes from different views are in different colors.

Figure 2.11. Incorrect elevations lead to distorted reconstruction. Our elevation estimation module can predict an accurate elevation of the input view.

2.3.4 Text to 3D Mesh

As shown in Figure 2.12, by integrating with off-the-shelf text-to-image 2D diffusion models [236, 247], our method can be naturally extended to support text-to-image-3D tasks and generate high-quality textured meshes in a short time.

2.4 Failure Cases and Limitations

Our method relies on Zero123 for generating multi-view images, which introduces challenges due to its occasional production of inconsistent results. In Figure 2.13, we present two typical cases that exemplify such inconsistencies. The first case involves an input view that lacks sufficient information, such as the back view of a fox. In this scenario, Zero123 struggles to generate consistent predictions for the invisible regions, such as the face of the fox. As a

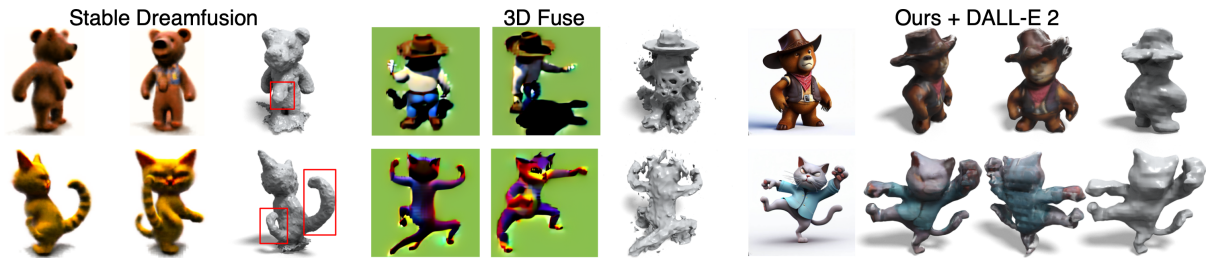


Figure 2.12. Text to 3D. First row: "a bear in cowboy suit." Second row: "a kungfu cat." We utilize DALL-E 2 [236] to generate an image conditioned on the text and then lift it to 3D. We compare our method with Stable Dreamfusion [225] and 3DFuse [257]. For baselines, volume renderings are shown.

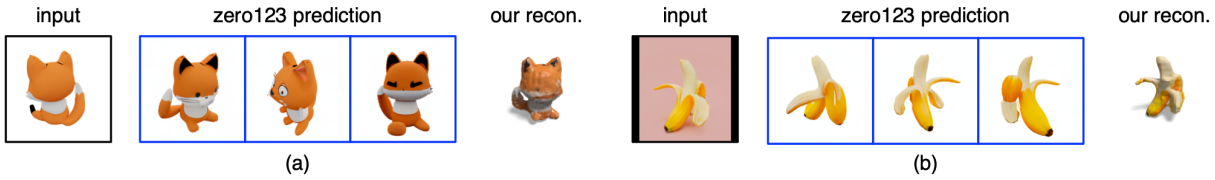


Figure 2.13. Failure cases of One-2-3-45. Our method relies on Zero123 to generate multi-view images, and we encounter challenges when Zero123 generates inconsistent results. (a) The input view lacks sufficient information. (b) The input view contains ambiguous or complicated structures.

consequence, our method may encounter difficulties in accurately inferring the geometry for those regions. The second case involves an input view with ambiguous or complex structures, such as the pulp and peel of a banana. In such situations, Zero123's ability to accurately infer the underlying geometry becomes limited. As a result, our method may be affected by the inconsistent predictions generated by Zero123. It is important to acknowledge that these limitations arise from the occasional scenarios, and they can impact the performance of our method in certain cases. Addressing these challenges and refining the reliability of Zero123's predictions remain areas for further investigation and improvement.

We have also noticed slight artifacts on the back side of our generated results. As one of the first works in combining view-conditioned 2D diffusion models with generalizable multi-view reconstruction, we believe that there is still ample room for exploring more advanced reconstruction techniques and incorporating additional regularizations. By doing so, we expect to significantly mitigate the minor artifacts and further enhance results in the future.

2.5 Summary

In this chapter, we present a novel method for reconstructing a high-quality 360° mesh of any object from a single image of it. In comparison to existing zero-shot approaches, our results exhibit superior geometry, enhanced 3D consistency, and a remarkable adherence to the input image. Notably, our approach reconstructs meshes in a single forward pass without the need for time-consuming optimization, resulting in significantly reduced processing time. Furthermore, our method can be effortlessly extended to support the text-to-3D task.

Chapter 2 incorporates material from the publication "One-2-3-45: Any Single Image to 3D Mesh in 45 Seconds without Per-Shape Optimization", by Minghua Liu, Chao Xu, Haian Jin, Linghao Chen, Mukund Varma T, Zexiang Xu, and Hao Su, published in *Advances in Neural Information Processing Systems (NeurIPS 2023)*. The dissertation author was primary investigator and the lead author of this paper.

Chapter 3

Open-World 3D Generation: Battling Multi-View Inconsistency

Although One-2-3-45 is one of the first single-image-to-3D methods to simultaneously address the challenges of generalizability and speed, producing 3D shapes in a single forward pass in just 45 seconds, its efficacy is often limited by the inconsistent multi-view predictions of Zero123, leading to compromised 3D reconstruction results, as shown in Section 2.4.

In this chapter, we introduce One-2-3-45++, a novel method that effectively overcomes the shortcomings of One-2-3-45, delivering significantly enhanced robustness and quality. Taking a single image of any object as input, One-2-3-45++ also includes two primary stages: 2D multi-view generation and 3D reconstruction. During the initial phase, rather than employing Zero123 to predict each view independently, One-2-3-45++ predicts consistent multi-view images jointly. This is realized by tiling a concise set of six-view images into a single image and then finetuning a 2D diffusion model to generate this combined image conditioned on the input reference image. In this way, the 2D diffusion net is able to attend to each view during generation, ensuring more consistent results across views. In the second stage, One-2-3-45++ employs a multi-view conditioned 3D-diffusion-based module to predict the textured mesh in a coarse-to-fine fashion. The consistent multi-view conditional images act as a blueprint for 3D reconstruction, facilitating a zero-shot hallucination capability. Concurrently, the 3D diffusion network excels in lifting the multi-view images, thanks to its ability to harness a broad spectrum of priors extracted from the

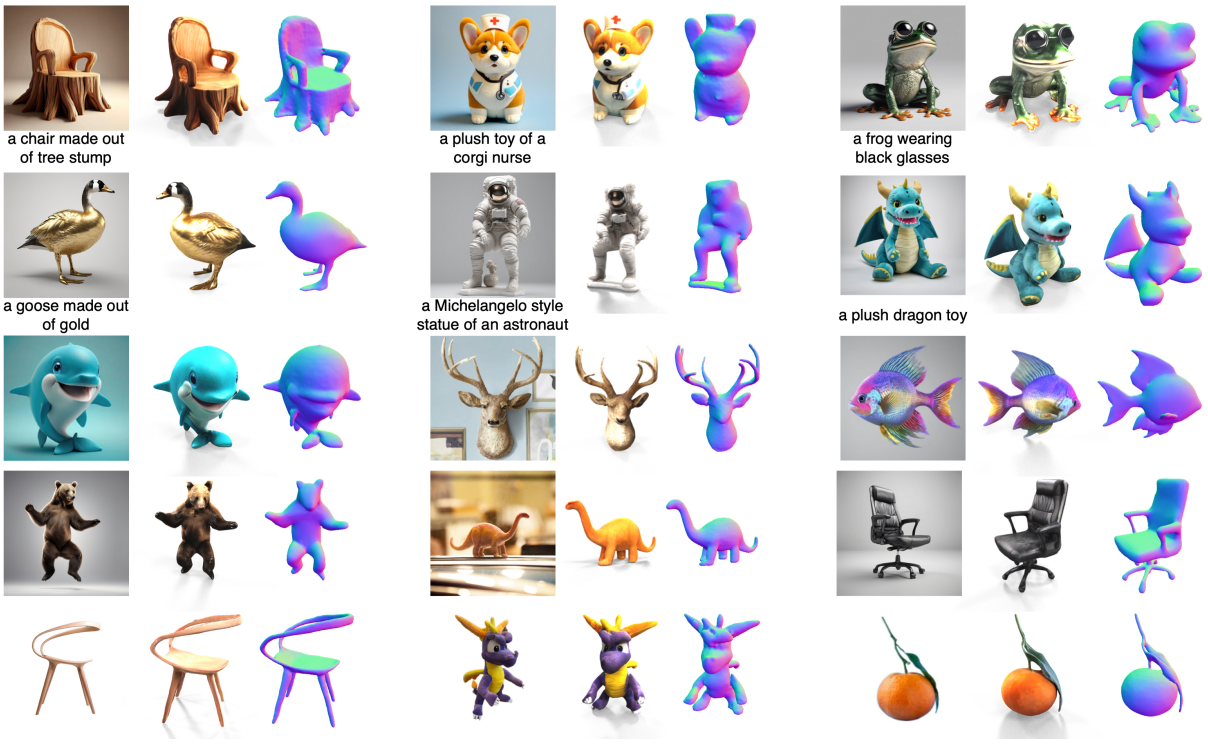


Figure 3.1. One-2-3-45++ is capable of transforming a single RGB image of any object into a high-fidelity textured mesh **in under one minute**. The generated meshes closely mirror the input image. Input image (and text prompt), textured mesh, and normal map are shown.

3D dataset. Ultimately, One-2-3-45++ employs a lightweight optimization technique to enhance the texture quality efficiently, leveraging the consistent multi-view images for supervision.

As depicted in Fig. 3.1, One-2-3-45++ efficiently generates 3D meshes with realistic textures in under a minute, offering precise fine-grained control. Our comprehensive evaluations, including user studies and objective metrics across an extensive test set, highlight One-2-3-45++'s superiority in terms of robustness, visual quality, and, most importantly, fidelity to the input image.

3.1 Related Work

3.1.1 3D Generation

3D generation has garnered significant attention in recent years. Before the advent of large-scale pre-trained 2D models, researchers often delved into 3D native generative models that

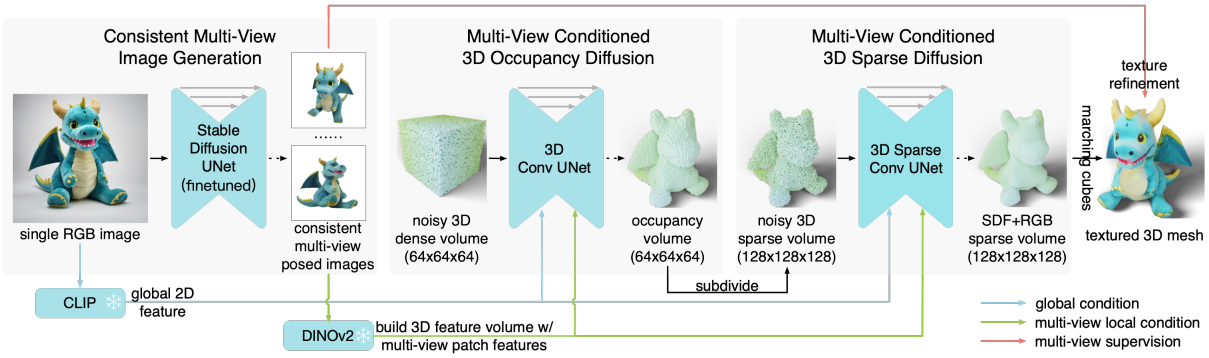


Figure 3.2. Starting with a single RGB image as input, we initially produce consistent multi-view images by fine-tuning a 2D diffusion model. These multi-view images are then elevated into 3D through a pair of 3D native diffusion networks. Throughout the 3D diffusion process, the generated multi-view images act as essential guiding conditions. After extracting the 3D mesh from the denoised volume, we further enhance the texture by employing a lightweight optimization with multi-view images as supervision. One-2-3-45++ is capable of producing an initial textured mesh **within 20 seconds** and delivering a refined one in **roughly one minute** using a single A100 GPU.

learn directly from 3D synthetic data or real scans and generate various 3D representations such as point clouds [4, 62, 197, 215, 373], 3D voxels [43, 252, 329, 335], polygon meshes [67, 78, 123, 163, 178, 213, 300], parametric models [96], and implicit fields [38, 61, 84, 121, 145, 199, 222, 327, 345, 370, 375, 390, 392]. However, given the limited availability of 3D data, these models tended to focus on a select number of categories (e.g., chairs, cars, planes, humans, etc.), struggling to generalize to unseen categories in the open world.

The advent of recent 2D generative models (e.g., DALL-E [237], Imagen [249], and Stable Diffusion [247]) and vision-language models (e.g., CLIP [234]) has equipped us with powerful priors about our 3D world, consequently fueling a surge of research in 3D generation. Notably, models like DreamFusion [225], Magic3D [154], and ProlificDreamer [314] have pioneered a line of approach for per-shape optimization [29, 36, 51, 110, 139, 196, 200, 201, 207, 232, 235, 257, 278, 279, 297, 340, 342, 368]. These models are designed to optimize a 3D representation for each unique input text or image, drawing on the 2D prior models for gradient guidance. While they have yielded impressive results, these methods tend to suffer from prolonged optimization times, the ”multi-face problem,” oversaturated colors, and a lack

of diversity in results. Some works also concentrate on creating textures or materials for input meshes, utilizing the priors of 2D models [27, 245].

A new wave of studies, highlighted by works like Zero123 [170], has showcased the promise of using pre-trained 2D diffusion models for synthesizing novel views from singular images or texts, opening new doors for 3D generation. For instance, One-2-3-45 [164], using multi-view images predicted by Zero123, can produce a textured 3D mesh in a mere 45 seconds. Nevertheless, the multi-view images produced by Zero123 lack 3D consistency. Our research, along with several concurrent studies [175, 183, 264, 323, 363], is dedicated to enhancing the consistency of these multi-view images – a vital step for subsequent 3D reconstruction applications.

3.1.2 Sparse View Reconstruction

While traditional 3D reconstruction methods, such as multi-view stereo or NeRF-based techniques, often demand a dense collection of input images for accurate geometry inference, many of the latest generalizable NeRF solutions [26, 119, 135, 176, 184, 243, 287, 301, 305, 356] strive to learn priors across scenes. This enables them to infer NeRF from a sparse set of images and generalize to novel scenes. These methods typically ingest a few source views as input, leveraging 2D networks to extract 2D features. These pixel features are then unprojected and aggregated into 3D space, facilitating the inference of density (or SDF) and colors. However, these methods may either rely on consistent multi-view images with accurate correspondences or possess limited priors to generalize beyond training datasets.

Recently, some methods [21, 124, 280, 395] have employed diffusion models to aid sparse view reconstruction tasks. However, they generally frame the problem as novel view synthesis, necessitating additional processing, such as distillation using a 3D representation, to generate 3D content. Our work utilizes a multi-view conditioned 3D diffusion model for 3D generation. This model directly learns priors from 3D data and obviates the need for additional post-processing. Moreover, some concurrent works [175, 183, 264] employ NeRF-based per-scene optimization

for reconstruction, leveraging specialized loss functions.

3.2 Proposed Method: One-2-3-45++

In traditional game studios, the creation of 3D content encompasses a series of stages, including concept art, 3D modeling, and texturing, etc. Each stage demands distinct and complementary expertise. For instance, concept artists should possess creativity, a vivid imagination, and the skill to visualize 3D assets. In contrast, 3D modelers must be skilled in 3D modeling tools and capable of interpreting and translating multi-view concept drawings into life-like models, even when drawings contain inconsistencies or errors.

One-2-3-45++ aims to harness the rich 2D priors and the valuable yet limited 3D data following a similar philosophy. As shown in Fig. 3.2, with a single input image of an object, One-2-3-45++ starts by generating coherent multi-view images of the object. This is achieved by finetuning a pre-trained 2D diffusion model and acts akin to the role of a concept artist. These generated images are then input into a multi-view conditioned 3D diffusion model for 3D modeling. The 3D diffusion module, trained on extensive multi-view and 3D pairings, excels at converting multi-view images into 3D meshes. Finally, the produced meshes undergo a lightweight refinement module, guided by the multi-view images, to further enhance the texture quality.

3.2.1 Consistent Multi-View Generation

Recently, Zero123 has demonstrated the potential of fine-tuning a pretrained 2D diffusion network to incorporate camera view control, thereby synthesizing novel views of an object from a single reference image. While previous studies have employed Zero123 to generate multi-view images, they often suffer from inconsistencies across different views. This inconsistency arises because Zero123 models the conditional marginal distribution for each view in isolation, without considering inter-view communication during multi-view generation. In this work, we present an innovative method to produce consistent multi-view images, significantly benefiting downstream

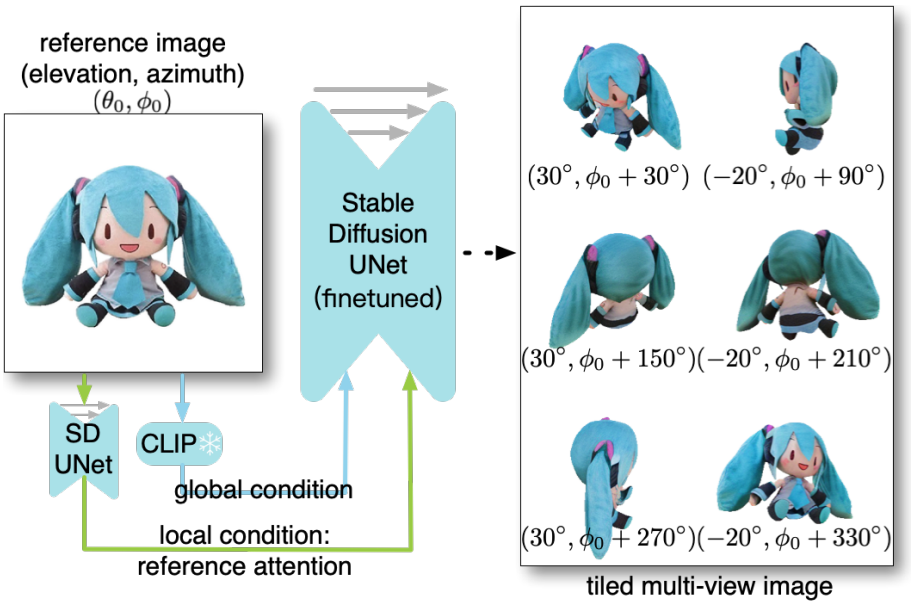


Figure 3.3. Consistent multi-view generation: We stitch multi-view images into a single frame and fine-tune the Stable Diffusion model to generate this composite image, using the input reference image as conditions. We utilize predetermined absolute elevation angles and relative azimuth angles. During 3D reconstruction, we do not need to infer the elevation angle of the input image.

3D reconstruction.

Multi-View Tiling To generate multiple views in a single diffusion process, we adopt a simple strategy by tiling a sparse set of 6 views into a single image with a 3×2 layout as shown in Fig. 3.3. Subsequently, we finetune a pre-trained 2D diffusion net to generate the composite image, conditioned on a single input image. This strategy enables multiple views to interact with each other during the diffusion.

It's nontrivial to define the camera poses of the multi-view images. Given that the 3D shapes within the training dataset lack aligned canonical poses, employing absolute camera poses for the multi-view images could lead to ambiguities for the generative model. Alternatively, if we were to set the camera poses relative to the input view, as done in Zero123, downstream applications would then be required to infer the elevation angle of the input image to deduce the camera poses of the multi-view images. This additional step could introduce errors into the pipeline. To address these, we opt for fixed absolute elevation angles paired with relative

azimuth angles to define the poses of multi-view images, effectively resolving the orientation ambiguity without necessitating further elevation estimation. To be more precise, the six poses are determined by alternating elevations of 30° and -20°, coupled with azimuths commencing at 30° and incrementing by 60° for each subsequent pose, as shown in Fig. 3.3.

Network and Training Details To fine-tune Stable Diffusion for adding image conditioning and generating coherent multi-view composite images, we employ three crucial network or training designs: (a) **Local Condition**: We adopt the reference attention technique [382] to incorporate the local condition of the image patch features. Specifically, we process the reference input image with the denoising UNet model and append the self-attention key and value matrices of the image tokens from the conditional reference image to the corresponding attention layers of the denoising multi-view image. (b) **Global Condition**: We leverage CLIP image embedding as a global condition, by replacing the text token features originally used in Stable Diffusion with the duplicated CLIP image features. These global image embeddings are multiplied by a set of learnable weights, providing the network with an overall semantic understanding of the object. (c) **Noise Schedule**: The original Stable Diffusion model was trained using a scaled-linear noise schedule. We found it necessary to switch to a linear noise scheme in our fine-tuning process.

We fine-tune the Stable Diffusion2 *v*-mode using 3D shapes from the Objaverse [50] dataset. For each shape, we generate three data points by randomly sampling the camera pose of the input image from a specified range, and selecting a random HDRI environment lighting from a curated set that offers uniform lighting. Initially, we fine-tuned only the self-attention layers along with the key and value matrices of the cross-attention layers using LoRA [101]. Subsequently, we fine-tuned the entire UNet using a conservative learning rate. The finetuning process was conducted using 16 A100 GPUs and took approximately 10 days.

3.2.2 3D Diffusion with Multi-View Condition

While prior work utilizes generalizable NeRF methods for 3D reconstruction, it primarily depends on accurate local correspondence of multi-view images and possesses limited priors for

3D generation. This constrains their effectiveness in lifting intricate and inconsistent multi-view images generated by the 2D diffusion network. Instead, we propose an innovative way to lift the generated multi-view images to 3D by utilizing a multi-view conditioned 3D generative model. It seeks to learn a manifold of plausible 3D shapes conditioned on multi-view images by training expressive 3D native diffusion networks on extensive 3D data.

3D Volume Representations As shown in Fig. 3.2, we represent a textured 3D shape as two discrete 3D volumes, a signed distance function (SDF) volume, and a color volume. The SDF volume measures the signed distance from the center of each grid cell to the nearest shape surface, while the color volume captures the color of the closest surface points relative to the center of the grid cells. Additionally, we generate a discrete occupancy volume for the 3D shape, where each grid cell stores a binary occupancy based on whether the absolute value of its SDF is below a predefined threshold. The occupancy volume depicts the shell of the 3D shape.

Two-Stage Diffusion Capturing fine-grained details of 3D shapes necessitates the use of high-resolution 3D grids, which unfortunately entail substantial memory and computational costs. We thus follow LAS-Diffusion [392] to generate high-resolution volumes in a coarse-to-fine two-stage manner. Specifically, the initial stage generates a low-resolution (e.g., $n = 64$) full 3D occupancy volume $F \in \mathbb{R}^{n\times n\times n\times 1}$ to approximate the shell of the 3D shape. The second stage then focuses on the occupied shell region only and aims to generate a high-resolution (e.g., 128^3) four-channel sparse volume S, which predicts fine-grained SDF values and color for the sparsely occupied shell region.

We employ a separate diffusion network for each stage. For the first stage, normal 3D convolution is used within the UNet to produce the full 3D occupancy volume F, while for the second stage, we incorporate 3D sparse convolution [276] in the UNet to yield the 3D sparse volume S. Both diffusion networks are trained using the denoising loss [95]:

$$\mathscr{L}_{x_0} = \mathbb{E}_{\varepsilon\sim\mathscr{N}(0,I),t\sim\mathscr{U}(0,1)} \left\| f\left(x_t, t, c\right) - x_0 \right\|_2^2$$

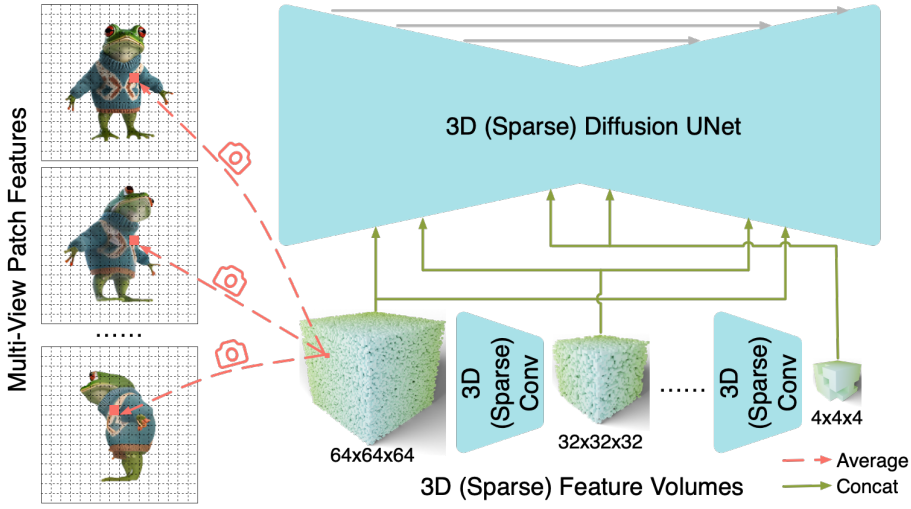


Figure 3.4. Multi-view local condition: We employ a pre-trained 2D backbone to extract 2D patch features for each view. These features are then aggregated using known projection matrices to construct a 3D feature volume. The volume is further processed by 3D convolutional neural networks, resulting in feature volumes of varying resolutions. Subsequently, these volumes are concatenated with the corresponding feature volumes within the diffusion U-Net to guide the 3D diffusion.

where ε and t are sampled noise and time step, x_0 is a data point (F or S) and x_t is its noised version, c is the multi-view condition, and f is the UNet. $\mathscr{N}$ and $\mathscr{U}$ denote Gaussian and uniform distribution, respectively.

Multi-View Condition Training a conventional 3D native diffusion network can be challenging to generalize due to the limited availability of 3D data. However, the use of generated multi-view images can provide a comprehensive guide, greatly simplifying the imagination difficulty of 3D generation. We integrate the multi-view images to guide the diffusion process by initially extracting local image features and subsequently constructing a conditional 3D feature volume, denoted as C. This strategy follows the rationale that local priors facilitate easier generalization [392].

As shown in Fig. 3.4, given m multi-view images, we first employ a pre-trained 2D backbone, DINOv2, to extract a set of local patch features for each image. We then build a 3D feature volume C by projecting each 3D voxel within the volume onto m multi-view images using the known camera poses. For each 3D voxel, we aggregate m associated 2D patch features through a shared-weight MLP, followed by max pooling. These aggregated features collectively

form the feature volume C.

In the diffusion network, the UNet consists of several levels. For example, the occupancy UNet in the initial stage has five levels: 64^3, 32^3, 16^3, 8^3, and 4^3. Initially, we construct a conditional feature volume C that matches the starting resolution, as outlined earlier. A 3D convolution network is then applied to C, producing volumes for the subsequent resolutions. The resultant conditional volumes are then concatenated with the volumes inside the UNet to guide the diffusion process. For the second stage, we construct sparse conditional volumes and utilize 3D sparse convolution. To benefit the diffusion of color volume, we also concatenate 2D pixel-wise projected colors to the final layer of the diffusion UNet. Moreover, we integrate the CLIP feature of the input image as a global condition.

Training and Inference Details We train the two diffusion networks using 3D shapes from the Objaverse dataset [50]. For each 3D shape, we first convert it to a watertight manifold before extracting its SDF volume. We unproject the multi-view renderings of the shape to get a 3D colored point cloud, which is used to build the color volume. During training, we utilize the ground truth renderings to serve as the multi-view conditions. Since two diffusion networks are trained separately, we introduced random perturbations to camera poses and infused random noises to the initial occupancy of the second stage to enhance robustness. We train the two diffusion nets using 8 A100 GPUs for about 10 days for each stage.

During inference, a 64^3 grid is first initialized with Gaussian noise and then denoised by the first diffusion net. Each predicted occupied voxel is further subdivided into 8 smaller voxels, used to construct a high-resolution sparse volume. The sparse volume is initialized with Gaussian noise and then denoised with the second diffusion net, resulting in predictions for the SDF and color of each voxel. The Marching Cubes algorithm is finally applied to extract a textured mesh.

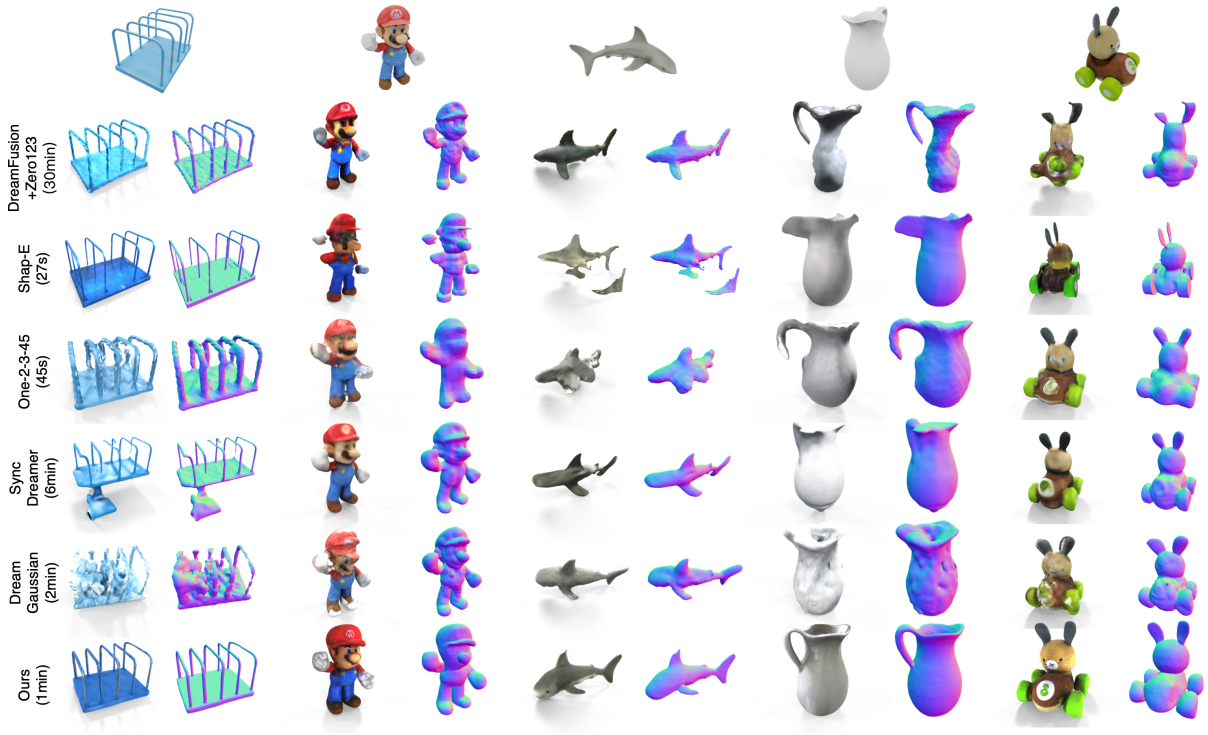


Figure 3.5. Qualitative results of various single image to 3D approaches. Input images, textured meshes, and normal maps are shown.

3.2.3 Texture Refinement

Given that multi-view images possess higher resolution than the 3D color volume, we can refine the texture of the generated mesh through a lightweight optimization process. To achieve this, we fix the geometry of the generated mesh while optimizing a color field represented by a TensoRF [25]. In each iteration, the generated 3D mesh is rasterized, and the color network is queried to produce 2D renderings. We leverage the predicted consistent multi-view images to guide the texture optimization using a $l2$ loss. Lastly, we bake the optimized color field onto the mesh, with the surface normal serving as the viewing direction.

Table 3.1. Comparison on single image to 3D. Evaluated on the GSO [59] dataset, which contains 1,030 3D objects.

Method	F-Sco. (%)↑	CLIP-Sim↑	User-Pref. (%)↑	Time↓
Zero123 XL [49]	91.6	73.1	58.6	30min
One-2-3-45 [164]	90.4	70.8	52.7	45s
SyncDreamer [175]	84.8	68.9	28.4	6min
DreamGaussian [278]	81.0	68.4	31.5	2min
Shap-E [121]	91.8	73.1	40.8	27s
Ours	**93.6**	**81.0**	**87.6**	60s

3.3 Experiments

3.3.1 Comparison on Image to 3D

Baselines: We evaluate One-2-3-45++ against both optimization-based and feed-forward methods. Within the optimization-based approaches, our baselines include DreamFusion [225] with Zero123 XL [170] as its backbone, as well as SyncDreamer [175], and DreamGaussian [278]. For feed-forward approaches, we compare with One-2-3-45 [164] and Shap-E [121]. We employ the ThreeStudio [82] implementation for Zero123 XL [82] and the original official implementations for the other methods.

Dataset and Metrics: We assess the performance of the methods using the entire set of 1,030 shapes from the GSO dataset [59], which were not exposed to any of the methods during training to the best of our knowledge. For each shape, we generate a frontal view image to serve as the input. In line with One-2-3-45 [164], we employ the F-Score and CLIP similarity as our evaluation metrics. The F-Score evaluates the geometric similarity between the predicted mesh and the ground truth mesh. For the CLIP similarity metric, we render 24 different views for each predicted and ground truth mesh, compute the CLIP similarity for each corresponding pair of images, and then average these values across all views. Prior to metric computation, we align the predicted mesh with the ground truth mesh using a combination of linear search and the ICP algorithm.

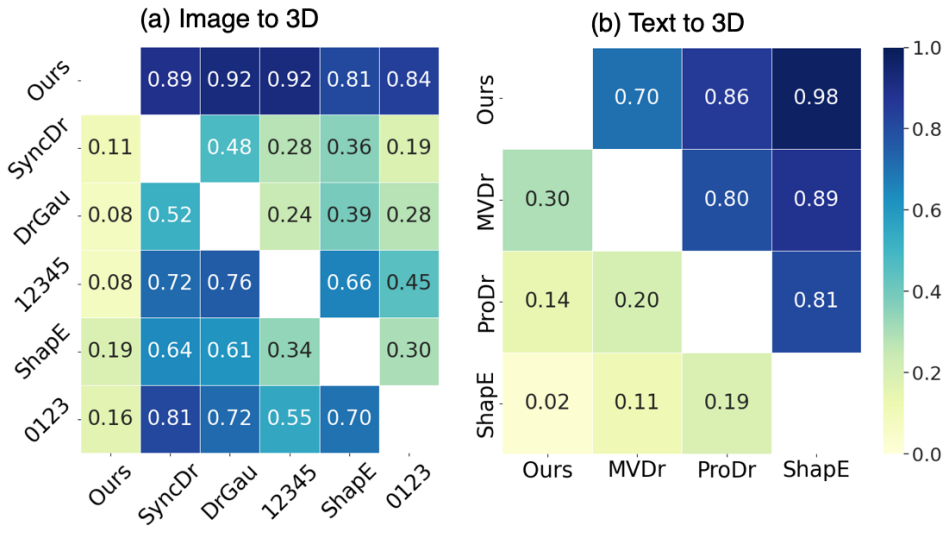


Figure 3.6. Results of a user study involving 53 participants. Each cell displays the probability or preference rate at which one method (row) outperforms another (column).

User Study: A user study was also carried out. For each participant, 45 shapes were randomly selected from the entire GSO dataset, and two methods were randomly sampled for each shape. Participants were asked to choose the result from each pair of comparative outcomes that exhibits superior quality and better aligns with the input image. The preference rate for all methods was then tallied based on these selections. In total, 2,385 evaluated pairs were collected from 53 participants.

Results: As presented in Tab. 3.1, One-2-3-45++ surpasses all baseline methods regarding F-Score and CLIP similarity. The user preference scores further highlight a significant performance disparity, with our method outperforming competing approaches by a substantial margin. Refer to Fig. 3.6 for an in-depth confusion matrix, which illustrates that One-2-3-45++ outperforms One-2-3-45 92% of the time. Moreover, when compared to optimization-based methods, our approach demonstrates notable runtime advantages. Fig. 3.5 and 3.7 show qualitative results.

3.3.2 Comparison on Text to 3D

Baselines: We compared One-2-3-45++ with optimization-based methods, specifically ProlificDreamer [314] and MVDream [264], as well as a feed-forward approach, Shap-E [121].

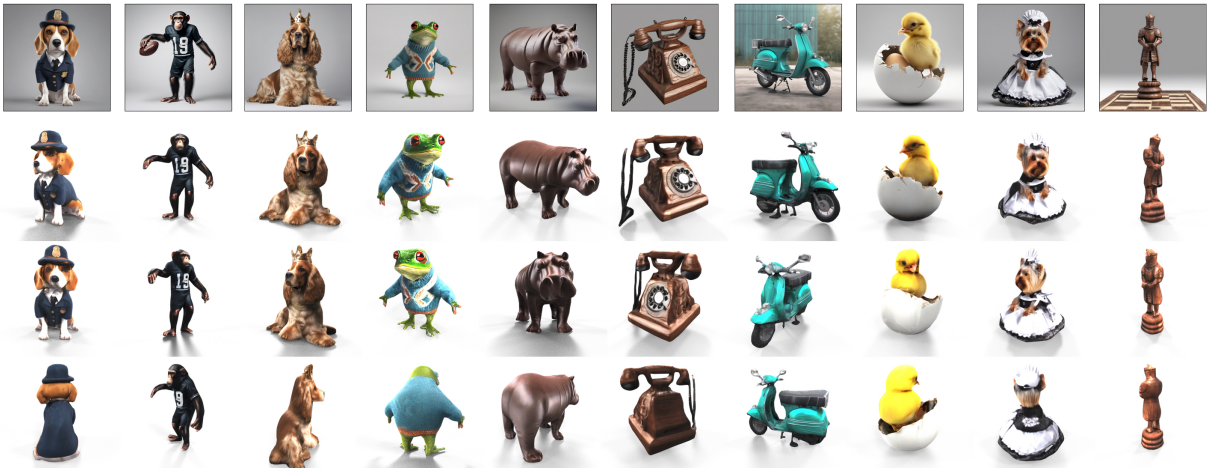


Figure 3.7. Our qualitative results: top row displays input images; subsequent rows show multi-view renderings of the generated meshes.

Figure 3.8. Qualitative results of various text to 3D approaches. Input images, textured meshes, and normal maps are shown.

For ProlificDreamer, we utilized the ThreeStudio implementation [82], while for the remaining methods, we employed their respective official implementations.

Dataset and Metrics: Given that many baseline approaches necessitate hours to produce a single 3D shape, our evaluation was conducted on 50 text prompts, sampled from DreamFusion [225]. We utilize CLIP similarity, calculated by comparing 24 rendered views of the predicted mesh against the input text prompt and then averaging the similarity scores across all views.

User Study: The user study, akin to the image-to-3D evaluation, involved 30 pairs of outcomes randomly selected for each participant. In total, 1,590 evaluation pairs were collected from 53

Table 3.2. Quantitative comparison with various text to 3D methods. Evaluated on 50 text prompts from DreamFusion [225].

Method	CLIP-Sim↑	User-Pref.↑	Runtime↓
ProlificDreamer [314]	25.7	39.5	10h+
MVDream [264]	24.8	66.2	2h
Shap-E [121]	22.3	11.1	27s
Ours	**26.8**	**84.1**	60s

participants.

Results: As illustrated in Tab. 3.2, One-2-3-45++ outperforms all baseline methods in terms of CLIP similarity. This is further corroborated by user preference scores, with our method significantly outshining rival techniques. See Fig. 3.6 for an in-depth analysis. When directly comparing One-2-3-45++ with the second-best method, MVDream [264], our approach commands a 70% user preference rate. Moreover, while our method delivers prompt results, MVDream [264] requires about 2 hours to generate a single shape. Fig. 3.8 shows qualitative results.

3.3.3 Analyses

Ablation Studies of Overall Pipeline One-2-3-45++ is comprised of three key modules: consistent multi-view generation, multi-view conditioned 3D diffusion, and texture refinement. We conducted ablation studies on these modules using the complete GSO dataset [59], with results detailed in Tab. 3.3. Replacing our consistent multi-view generation module with Zero123XL [49] led to a noticeable performance decline. Furthermore, substituting our 3D diffusion module with the generalizable NeRF used in One-2-3-45 [164] resulted in an even more significant performance drop. However, the inclusion of our texture refinement module markedly improved texture quality, yielding higher CLIP similarity scores.

Ablation Studies of 3D Diffusion Tab. 3.4 presents the results of an ablation study of the 3D diffusion module. The study highlights the importance of multi-view images for the module's efficacy. When the module operates without multi-view conditions, relying solely on the global CLIP feature from a single input view (rows a and f), there is a significant decline in performance.

Table 3.3. Ablation studies of different modules. Evaluated on the complete GSO [59] dataset. "MultiView", "Reconstruction", and "Texture" indicate multi-view generation, sparse view reconstruction, and texture refinement modules, respectively.

MultiView	Reconstruction	Texture	F-Sc.↑	CLIP-Sim↑	Time↓
Zero123 XL [49]	Ours	w/o	92.9	71.9	14s
Ours	SparseNeuS [184]	w/o	81.2	67.2	15s
Ours	Ours	w/o	**93.6**	73.4	20s
Ours	Ours	w/	**93.6**	**81.0**	60s

Table 3.4. Ablation study of the 3D diffusion module. 3D IoU of the initial-stage occupancy prediction is reported. Note that the 3D IoU is computed for the 3D shell, excluding the solid interior.

id	multi-view cond.	global cond.	image source	proj. perturb.	3D IoU ↑
a	w/o	w/	rendering	N/A	18.3
b	global	w/	rendering	N/A	24.4
c	local	w/o	rendering	w/o	41.4
d	local	w/	prediction	w/o	41.9
e	local	w/	rendering	w/o	44.1
f	local	w/	rendering	w/	45.1

Conversely, the One-2-3-45++ approach leverages multi-view local features by constructing a 3D feature volume with known projection matrices. A mere concatenation of global CLIP features from multiple views also impairs performance (rows b and f), underlining the value of multi-view local conditions. Global CLIP features of the input view, however, provide global shape semantics; their removal results in decreased performance (rows c and e). Although One-2-3-45++ uses predicted multi-view images for 3D reconstruction, incorporating these predicted images during training of the 3D diffusion module can lead to a performance downturn (rows d and e) due to the potential mismatch between the predicted multi-view images and actual 3D ground truth meshes. To train the module effectively, we utilize ground truth renderings. Recognizing that predicted multi-view images may be flawed, we introduce random perturbations to projection matrices during training to enhance robustness when processing predicted multi-view images (rows e and f).

Table 3.5. Comparison of different multi-view generation methods. Evaluated on the complete GSO [59] dataset.

	Target Elevations	PSNR ↑	LPIPS ↓	Mask IoU ↑
Zero123 [170]	30° and −20°	20.32	**0.110**	0.856
Zero123 XL [49]		20.11	0.113	0.869
Ours		**22.12**	**0.110**	**0.878**
SyncDreamer [175]	30°	21.67	0.095	0.894
Wonder3D [183]	0°	18.67	0.130	0.635

Comparison on Multi-View Generation We also evaluate our consistent multi-view generation module against existing approaches, namely Zero123 [170] and its scaled variant [49], alongside two concurrent works: SyncDreamer [175] and Wonder3D [183]. Our comparison utilizes the GSO [59] dataset, where for each object, we render a single input image and task the methods with producing multi-view images. For Zero123 and Zero123 XL, we utilize the same target poses as our approach. However, for Wonder3D and SyncDreamer, we employ the target poses preset by these methods, as they do not support altering camera positions during inference. As presented in Tab. 3.5, our approach surpasses current methodologies in PSNR, LPIPS, and foreground mask IoU. Notably, Wonder3D [183] employs orthographic projection in its training phase, which compromises its robustness when dealing with perspective images during inference. SyncDreamer [175] only generates views at an elevation of 30°, a simpler setting than ours.

3.4 Summary

In this chapter, we introduced One-2-3-45++, an innovative approach for transforming a single image of any object into a 3D textured mesh. This method stands out by offering more precise control compared to existing text-to-3D models, and it is capable of delivering high-quality meshes swiftly—typically in under 60 seconds. Additionally, the generated meshes exhibit a high fidelity to the original input image. Looking ahead, there is potential to enhance the robustness and detail of the geometry by incorporating additional guiding conditions from 2D diffusion models, alongside RGB images.

Figure 3.9. SpaRP handles open-world 3D reconstruction and pose estimation from unposed sparse-view images, delivering results in approximately 20 seconds.

3.5 Extension: Reconstruction and Pose Estimation from Sparse Views

In this chapter and the last, we introduced methods for converting a single image to 3D and text to 3D. While these methods can achieve high-quality geometry and texture that match the input view, they also introduce ambiguities in regions not visible in the input image, such as the back view. Although these methods attempt to generate plausible interpretations of these unseen areas, the resulting regions may not always align with users' expectations, and users

Figure 3.10. Single-View vs. Sparse-View Input for 3D Reconstruction. We compare the results of SpaRP when using single-view and sparse-view inputs.

often lack sufficient control over these ambiguous regions, as shown in Figure 3.10.

In this section, we explore a critical scenario where the input consists of one or a few unposed 2D images of a single object. The images are captured from arbitrarily distributed camera poses, often with little to no overlap. We tackle both the 3D reconstruction and pose estimation of input images under this sparse view setting. Note that, in dense view setting, traditional Structure-from-Motion (SfM) solvers (e.g., COLMAP [255]) are typically employed for pose estimation. However, with sparse view inputs, these solvers often become unreliable and tend to fail due to insufficient overlapping visual cues. This issue is the main reason why existing sparse view reconstruction methods [130, 184, 395] generally require known camera poses as input. While some recent methods have attempted pose-free reconstruction and pose estimation for sparse views [116, 117, 153, 266, 378], they are usually trained on a predefined small set of object categories and exhibit poor generalization to unseen object categories.

In response, we propose an innovative class-agnostic approach called SpaRP [339], capable of processing arbitrary object categories with unposed sparse views. Our inspiration comes from recent breakthroughs in open-domain single-image-to-3D methods. They leverage 2D diffusion models (e.g., Stable Diffusion [247]) to generate novel viewpoints of an object [170],

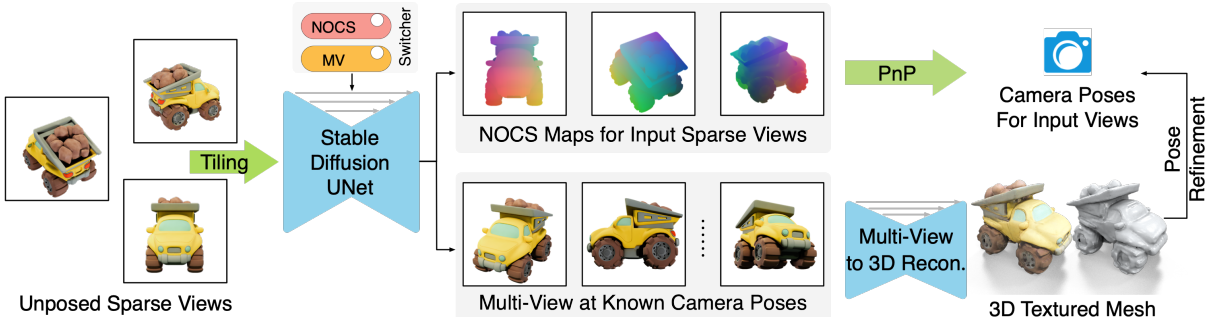


Figure 3.11. Pipeline Overview of SpaRP. We begin by taking a sparse set of unposed images as input, which we tile into a single composite image. This composite image is subsequently provided to the Stable Diffusion UNet to serve as the conditioning input. The 2D diffusion model is simultaneously finetuned to predict NOCS maps for the input sparse views and multi-view images under known camera poses. From the NOCS maps, we extract the camera poses corresponding to the input views. The multi-view images are then processed by a reconstruction module to generate textured 3D meshes. Optionally, the camera poses can be further refined using the generated mesh for improved accuracy.

and even consistent multi-view images from a single input image [141, 175, 183, 262, 264], by finetuning the diffusion models with corresponding multi-view image pairs. These discoveries imply that 2D diffusion models harbor rich priors concerning 3D objects. Instead of merely producing multi-view images, we contemplate leveraging 2D diffusion models to examine a set of unposed input images from sparse viewpoints, infer their spatial interrelationships, and recover relative camera poses and underlying 3D shapes.

As shown in Figure 3.11, we finetune a 2D diffusion model [247] to process sparse input views by compositing them into a single image for conditioning. The diffusion model is concurrently tuned to deduce the relative poses of the input images and the underlying 3D objects. For the relative pose estimation branch, instead of outputting camera poses as scalars, we task 2D diffusion models to produce a surrogate representation: the NOCS maps [298] that embed pixel-wise correspondences across different views and are more suitable for 2D diffusion models. From these maps, we extract the relative camera poses for the sparse views using the traditional PnP algorithm [18], assuming known camera intrinsics. For the reconstruction branch, the diffusion model is tasked to produce multi-view images of the object from fixed known

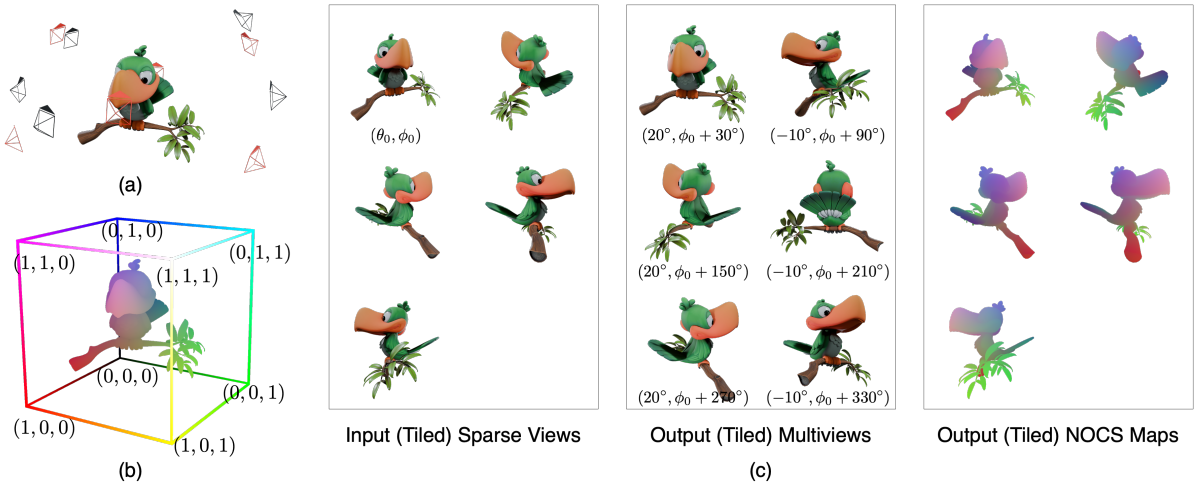


Figure 3.12. Illustration of the input and output of the multi-view diffusion model in SpaRP. (a) Regardless of the poses of the sparse input views (in black), the output multiviews are uniformly distributed (in red) and encompass the entire 3D object. (b) The Normalized Object Coordinate Space (NOCS) of the object, whose orientation is aligned with the azimuth of the first input view. (c) An example of input and output tiled images. The elevation and azimuth of the first input view are denoted by θ_0 and ϕ_0, respectively. The camera poses of the output multiview images are determined by ϕ_0. The output NOCS maps correspond to the input sparse views, and the orientation of the coordinate frame is also determined by ϕ_0.

camera poses, covering the entire 3D object. This task requires the models to incorporate all information from input sparse views and hallucinate invisible regions. We then feed the generated images with fixed known poses into a pre-trained 3D reconstruction module [161] to create a textured 3D mesh. We can further refine the estimated camera poses by aligning the input views with the generated mesh through differentiable rendering [136]. Please refer to Figure 3.12 for an illustration of the input and output of the multi-view diffusion model used in SpaRP.

We train SpaRP on the Objaverse [50] dataset with 1–6 unposed input views. Unlike some previous methods that rely on costly per-shape optimization [328], our method delivers 3D textured meshes along with camera poses in a much more efficient manner, requiring only $\sim$16 seconds. As shown in Figure 3.9, our approach can faithfully generate 3D assets that closely follow the reference unposed images, effectively overcoming the ambiguity issue of single-image-to-3D. Extensive evaluation on three datasets demonstrates the superior performance of our method over baselines in reconstructing 3D meshes with vivid appearance and high-fidelity

geometry, alongside precise pose estimation of the input images.

Chapter 3 incorporates material from the publication “One-2-3-45++: Fast Single Image to 3D Objects with Consistent Multi-View Generation and 3D Diffusion”, by Minghua Liu, Ruoxi Shi, Linghao Chen, Zhuoyang Zhang, Chao Xu, Xinyue Wei, Hansheng Chen, Chong Zeng, Jiayuan Gu, and Hao Su, published in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2024)*. The dissertation author was primary investigator and the lead author of this paper.

Chapter 4

Open-World 3D Generation: The Magic of 3D Native Guidance

In the previous chapters, we introduced a promising strategy for open-world 3D generation, which involves first predicting a sparse set of multi-view images using 2D diffusion models [170, 262], and then lifting these predicted images into a 3D model by training a feed-forward network [161, 164]. As this strategy effectively addresses both the speed and generalizability challenges, it has gradually garnered significant attention and many followers in the community.

While many recent works explore utilizing priors from 2D diffusion models, such as generating consistent multi-view images [262, 264] and predicting normal maps from RGB [64, 183, 262], the feed-forward model that converts multi-view images into 3D remains underexplored. One-2-3-45 [164] leverages a generalizable NeRF method for 3D reconstruction but suffers from limited quality and success rates. One-2-3-45++ [161] improves on this by using a two-stage 3D diffusion model, yet it still struggles to generate high-quality textures or fine-grained geometry. Given that sparse-view reconstruction of open-world objects requires extensive priors, another family of works pioneered by the large reconstruction model (LRM) [97] combines large-scale transformer models with the triplane representation and trains the model primarily using rendering loss. Although straightforward, these methods typically require over a hundred GPUs to train. Moreover, due to their reliance on volume rendering, these methods have

Figure 4.1. Given a sparse set (e.g., 6) of multi-view RGB images and their normal maps as input, MeshFormer reconstructs high-quality 3D textured meshes with fine-grained, sharp geometric details in a feed-forward pass of just a few seconds. Here, ground truth multi-view images are used as input.

difficulty extracting high-quality meshes. For instance, some recent follow-up works [320, 341] implement complex multi-stage "NeRF-to-mesh" training strategies, but the results still leave room for improvement.

In this chapter, we present MeshFormer [165], an open-world sparse-view reconstruction model that takes a sparse set of posed images of an arbitrary object as input and delivers high-quality 3D textured meshes with a single forward pass in a few seconds. Instead of representing 3D data as "2D planes" and training a "black box" transformer model optimizing only rendering loss, we find that by incorporating various types of 3D-native priors into the

model design, including network architecture, supervision signals, and input guidance, our model can significantly improve both mesh quality and training efficiency. Specifically, we propose representing features in explicit 3D voxels and introduce a novel architecture that combines large-scale transformers with 3D (sparse) convolutions. Compared to triplanes and pure transformers models with little 3D-native design, MeshFormer leverages the explicit 3D structure of voxel features and the precise projective correspondence between 3D voxels and 2D multi-view features, enabling faster and more effective learning.

Unlike previous works that rely on NeRF-based representation in their pipeline, we utilize mesh representation throughout the process and train MeshFormer in a unified, single-stage manner. Specifically, we propose combining surface rendering with additional explicit 3D supervision, requiring the model to learn a signed distance function (SDF) field. The network is trained with high-resolution SDF supervision, and efficient differentiable surface rendering is applied to the extracted meshes for rendering losses. Due to the explicit 3D geometry supervision, MeshFormer enables faster training while eliminating the need for expensive volume rendering and learning an initial coarse NeRF. Furthermore, in addition to multi-view posed RGB images, we propose using corresponding normal maps as input, which can be captured through sensors and photometric techniques [28, 326] or directly estimated by recent 2D vision models [64, 183, 262]. These multi-view normal images provide important clues for 3D reconstruction and fine-grained geometric details. We also task the model with learning a normal texture in addition to the RGB texture, which can then be used to enhance the generated geometry through a traditional post-processing algorithm [214].

Thanks to the explicit 3D-native structure, supervision signal, and normal guidance that we have incorporated, MeshFormer can generate high-quality textured meshes with fine-grained geometric details, as shown in Figure 4.1. Compared to concurrent methods that require over one hundred GPUs or complex multi-stage training, MeshFormer can be trained more efficiently and conveniently with just eight GPUs over two days, achieving on-par or even better performance. It can also seamlessly integrate with various 2D diffusion models to enable numerous tasks, such

as single-image-to-3D and text-to-3D. In summary, our key contributions include:

- We introduce MeshFormer, an open-world sparse-view reconstruction model capable of generating high-quality 3D textured meshes with fine-grained geometric details in a few seconds. It can be trained with only 8 GPUs, outperforming baselines that require over one hundred GPUs.
- We propose a novel architecture that combines 3D (sparse) convolution and transformers. By explicitly leveraging 3D structure and projective bias, it facilitates better and faster learning.
- We propose a unified single-stage training strategy for generating high-quality meshes by combining surface rendering and explicit 3D geometric supervision.
- We are the first to introduce multi-view normal images as input to the feed-forward reconstruction network, providing crucial geometric guidance. Additionally, we propose to predict extra 3D normal texture for geometric enhancement.

4.1 Related Work

Open-world 3D Object Generation Open-world 3D object generation have recently made significant advancements, thanks to the emergence of large-scale 3D datasets [49, 50] and the extensive priors learned by 2D models [234, 237, 247, 249]. Exemplified by DreamFusion [225], a line of work [29, 36, 51, 139, 154, 232, 257, 264, 271, 278, 297, 314] uses 2D models as guidance to generate 3D objects through per-shape optimization with SDS-like losses. Although these methods produce increasingly better results, they are still limited by lengthy runtimes and many other issues. Another line of work [97, 121, 186, 215, 337, 398] trains a feed-forward generative model solely on 3D data that consumes text prompts or single-image inputs. While fast during inference, these methods struggle to generalize to unseen object categories due to the scarcity of 3D data. More recently, works such as Zero123 [170] have shown that 2D diffusion models can be fine-tuned with 3D data for novel view synthesis. A line of

work [141, 141, 161, 277, 315, 320, 341], pioneered by One-2-3-45 [164], proposes first predicting multi-view images through 2D diffusion models and then lifting them to 3D through a feed-forward network, effectively addressing the speed and generalizability issues. Many recent works have also explored better strategies to fine-tune 2D diffusion models for enhancing the 3D consistency of multi-view images [69, 102, 133, 175, 177, 233, 262, 264, 292, 303, 323, 325, 363, 371]. In addition to the feed-forward models, the generated multi-view images can also be lifted to 3D through optimizations [69, 175, 183].

Sparse-View Feed-Forward Reconstruction Models When a small baseline between input images is assumed, existing generalizable NeRF methods [176, 242, 287, 356] aim to find pixel correspondences and learn generalizable priors across scenes by leveraging cost-volume-based techniques [26, 184, 367] or transformer-based structures [119, 135, 243, 301, 305]. Some of methods have also incorporated a 2D diffusion process into the pipeline [21, 124, 280]. However, these methods often struggle to handle large baseline settings (e.g., only frontal-view reconstruction) or are limited by a small training set and fail to generalize to open-world objects. Recently, many models [141, 277, 304, 315, 320, 341, 348, 349, 380, 391] specifically aimed at open-world 3D object generation have been proposed. They typically build large networks and aim to learn extensive reconstruction priors by training on large-scale 3D datasets [50]. For example, the triplane representation and transformer models are often used. By applying volume rendering or Gaussian splatting [277, 348, 380], they train the model with rendering losses. However, these methods typically require extensive GPUs to train and have difficulty extracting high-quality meshes. While some recent (concurrent) works [320, 341] utilize multi-stage "NeRF-to-mesh" training strategies to improve the quality, the results still leave room for improvement.

Geometry Guidance for 3D Reconstruction Many recent works have shown that in addition to multi-view RGB images, 2D diffusion models can be fine-tuned to generate other geometric modalities, such as depth maps [313], normal maps [64, 183, 187], or coordinate maps [146, 315]. These additional modalities can provide crucial guidance for 3D generation and

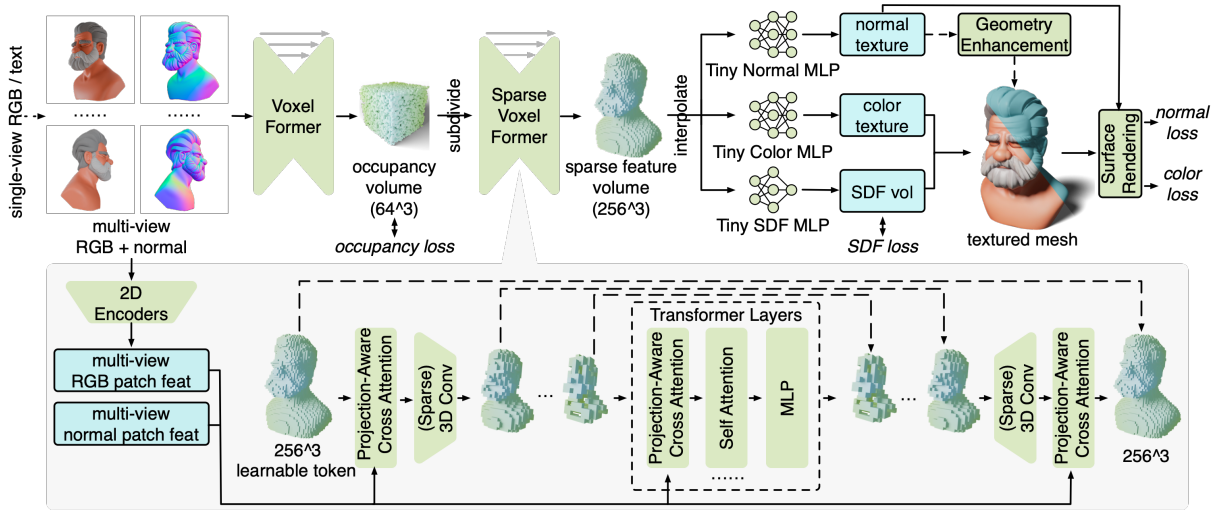


Figure 4.2. Pipeline Overview. MeshFormer takes a sparse set of multi-view RGB and normal images as input, which can be estimated using existing 2D diffusion models. We utilize a 3D feature volume representation, and submodules Voxel Former and Sparse Voxel Former share a similar novel architecture, detailed in the gray region. We train MeshFormer in a unified single stage by combining mesh surface rendering and 512^3 SDF supervision. MeshFormer learns an additional normal texture, which can be used to further enhance the geometry and generate fine-grained sharp geometric details.

reconstruction. While many recent methods utilize these geometric cues as inverse optimization guidance [29, 64, 146, 183, 233, 315], we propose to take normal maps as input in a feed-forward reconstruction model and task the model with generating 3D-consistent normal texture for geometry enhancement of sharp details.

4.2 Proposed Method: MeshFormer

As shown in Figure 4.2, MeshFormer takes a sparse set of posed multi-view RGB and normal images as input and generates a high-quality textured mesh in a single feed-forward pass. In the following sections, we will first introduce our choice of 3D representation and a novel model architecture that combines large-scale transformers with 3D convolutions (Sec. 4.2.1). Then, we will describe our training objectives, which integrate surface rendering and explicit 3D SDF supervision (Sec. 4.2.2). Last but not least, we will present our normal guidance and geometry enhancement module, which plays a crucial role in generating high-quality meshes

with fine-grained geometric details (Sec. 4.2.3).

4.2.1 3D Representation and Model Architecture

Triplane vs. 3D Voxels Open-world sparse-view reconstruction requires extensive priors, which can be learned through a large-scale transformer. Prior arts [141, 286, 315, 320, 341] typically utilize the triplane representation, which decomposes a 3D neural field into a set of 2D planes. While straightforward for processing by transformers, the triplane representation lacks explicit 3D spatial structures and makes it hard to enable precise interaction between each 3D location and its corresponding 2D projected pixels from multi-view images. For instance, these methods often simply apply self-attention across all triplane patch tokens and cross-attention between triplane tokens and all multi-view image tokens. This all-to-all attention is not only costly but also makes the methods cumbersome to train. Moreover, the triplane representation often shows results with notable artifacts at the boundaries of patches and may suffer from limited expressiveness for complex structures. Consequently, we choose the 3D voxel representation instead, which explicitly preserves the 3D spatial structures.

Combining Transformer with 3D Convolution To leverage the explicit 3D structure and the powerful expressiveness of a large-scale transformer model while avoiding an explosion of computational costs, we propose VoxelFormer and SparseVoxelFormer, which follow a 3D UNet architecture while integrating a transformer at the bottleneck. The overall idea is that we use local 3D convolution to encode and decode a high-resolution 3D feature volume, while the global transformer layer handles reasoning and memorizing priors for the compressed low-resolution feature volume. Specifically, as shown in Figure 4.2, a 3D feature volume begins with a learnable token shared by all 3D voxels. With the 3D voxel coordinates, we can leverage the projection matrix to enable each 3D voxel to aggregate 2D local features from multi-view images via a projection-aware cross-attention layer. By iteratively performing projection-aware cross-attention and 3D (sparse) convolution, we can compress the 3D volume to a lower-resolution one. After compression, each 3D voxel feature then serves as a latent token, and a deep transformer model

is applied to a sequence of all 3D voxel features (position encoded) to enhance the model's expressiveness. Finally, we use the convolution-based inverse upper branch with skip connection to decode a 3D feature volume with the initial high resolution.

Projection-Aware Cross Attention Regarding 3D-2D interaction, the input multi-view RGB and normal images are initially processed by a 2D feature extractor, such as a trainable DINOv2 [218], to generate multi-view patch features. While previous cost-volume-based methods [26, 184] typically use mean or max pooling to aggregate multi-view 2D features, these simple pooling operations might be suboptimal for addressing occlusion and visibility issues. Instead, we propose a projection-aware cross-attention mechanism to adaptively aggregate the multi-view features for each 3D voxel. Specifically, we project each 3D voxel onto the m views to interpolate m RGB and normal features. We then concatenate these local patch features with the projected RGB and normal values to form m 2D features. In the projection-aware cross-attention module, we use the 3D voxel feature to calculate a query and use both the 3D voxel feature and the m 2D features to calculate $m+1$ keys and values. A cross-attention is then performed for each 3D voxel, enabling precise interaction between each 3D location and its corresponding 2D projected pixels, and allowing adaptive aggregation of 2D features.

Coarse-to-Fine Feature Generation As shown in Figure 4.2, to generate a high-resolution 3D feature volume that captures the fine-grained details of 3D shapes, we follow previous work [161, 392] by employing a coarse-to-fine strategy. Specifically, we first use VoxelFormer, which is equipped with full 3D convolution, to predict a low-resolution, coarse 3D occupancy volume. Each voxel in this volume stores a binary value indicating whether it is close to the surface. The predicted occupied voxels are then subdivided to create higher-resolution sparse voxels. Next, we utilize a second module, SparseVoxelFormer, which features 3D sparse convolution [276], to predict features for these sparse voxels. After this, we can interpolate the 3D feature of any near-surface 3D point, which encodes both geometric and color information, from the high-resolution sparse feature volume. The features can then be fed into various MLPs to learn the corresponding fields.

4.2.2 Unified Single-Stage Training: Surface Rendering with SDF Supervision

Existing works typically use NeRF [202] and volume rendering or 3D Gaussian splatting [127] since they come with a relatively easy and stable learning process. However, extracting high-quality meshes from their results is often non-trivial. For example, directly applying Marching Cubes [185] to density fields of learned NeRFs typically generates meshes with many artifacts. Recent methods [319, 320, 341] have designed complex, multi-stage "NeRF-to-mesh" training with differentiable surface rendering, but the generated meshes still leave room for improvement. On the other hand, skipping a good initialization and directly learning meshes from scratch using purely differentiable surface rendering losses is also infeasible, as it is highly unstable to train and typically results in distorted geometry.

In this work, we propose leveraging explicit 3D supervision in addition to 2D rendering losses. As shown in Figure 4.2, we task MeshFormer with learning a signed distance function (SDF) field supervised by a high-resolution (e.g., 512^3) ground truth SDF volume. The SDF loss provides explicit guidance for the underlying 3D geometry and facilitates faster learning. It also allows us to use mesh representation and differentiable surface rendering from the beginning without worrying about good geometry initialization or unstable training, as the SDF loss serves as a strong regularization for the underlying geometry. By combining surface rendering with explicit 3D SDF supervision, we train MeshFormer in a unified, single-stage training process. As shown in Figure 4.2, we employ three tiny MLPs that take as input the 3D feature interpolated from the 3D sparse feature volume to learn an SDF field, a 3D color texture, and a 3D normal texture. We extract meshes from the SDF volume using dual Marching Cubes [254] and employ NVDiffRast [136] for differentiable surface rendering. We render both the multi-view RGB and normal images and compute the rendering losses, which consist of both the MSE and perceptual loss terms. As a result, our training loss can be expressed as:

$$\mathscr{L} = \lambda_1 \mathscr{L}_{\text{MSE}}^{\text{color}} + \lambda_2 \mathscr{L}_{\text{LPIPS}}^{\text{color}} + \lambda_3 \mathscr{L}_{\text{MSE}}^{\text{normal}} + \lambda_4 \mathscr{L}_{\text{LPIPS}}^{\text{normal}} + \lambda_5 \mathscr{L}_{\text{occ}} + \lambda_6 \mathscr{L}_{\text{SDF}} \tag{4.1}$$

where L_{occ} and L_{SDF} are MSE losses for occupancy and SDF volumes, and λ_i denotes the weight of each loss term. Note that we do not use mesh geometry to derive normal maps; instead, we utilize the learned normal texture from the MLP, which will be detailed later.

4.2.3 Fine-Grained Geometric Details: Normal Guidance and Geometry Enhancement

Without dense-view correspondences, 3D reconstruction from sparse-view RGB images typically struggles to capture geometric details and suffers from texture ambiguity. While many recent works [141, 320, 341] attempt to employ large-scale models to learn mappings from RGB to geometric details, this typically requires significant computational resources. Additionally, these methods are primarily trained using 3D data, but it's still uncertain whether the scale of 3D datasets is sufficient for learning such extensive priors. On the other hand, unlike RGB images, normal maps explicitly encode geometric information and can provide crucial guidance for 3D reconstruction. Notably, open-world normal map estimation has achieved great advancements. Many recent works [64, 183, 262] demonstrate that 2D diffusion models, trained on billions of natural images, embed extensive priors and can be fine-tuned to predict normal maps. Given the significant disparity in data scale between 2D and 3D datasets, it may be more effective to use 2D models first for generating geometric guidance.

Input Normal Guidance As shown in Figure 4.2, in addition to multi-view RGB images, MeshFormer also takes multi-view normal maps as input, which can be generated using recent open-world normal estimation models [64, 183, 262]. In our experiments, we utilize Zero123++ v1.2 [262], which trains an additional ControlNet [381] over the multi-view prediction model. The ControlNet takes multi-view RGB images, predicted by Zero123++, as a condition and

produces corresponding multi-view normal maps, expressed in the camera coordinate frame. Given these maps, MeshFormer first converts them to a unified world coordinate frame, and then treats them similarly to the multi-view RGB images, using projection-aware cross-attention to guide 3D reconstruction. According to our experiments (Sec. 4.3.4), the multi-view normal maps enable the networks to better capture geometry details, and thus greatly improve final mesh quality.

Geometry Enhancement While the straightforward approach of deriving normal maps from the learned mesh and using a normal loss to guide geometry learning has been commonly used, we find that this approach makes our mesh learning less stable. Instead, we propose learning a 3D normal texture, similar to a color texture, using a separate MLP. By computing the normal loss for MLP-queried normal maps instead of mesh-derived normal maps, we decouple normal texture learning from underlying geometry learning. This makes the training more stable, as it is easier to learn a sharp 3D normal map than to directly learn a sharp mesh geometry. The learned 3D normal texture can be exported with the mesh, similar to the color texture, to support various graphics rendering pipelines. In applications that require precise 3D geometry, such as 3D printing, the learned normal texture can also be used to refine the mesh geometry with traditional algorithms. Specifically, during inference, after extracting a 3D mesh from the SDF volume, we utilize a post-processing algorithm [214] that takes as input the 3D positions of the mesh vertices and the vertex normals estimated from the MLP. The algorithm adjusts the mesh vertices to align with the predicted normals in a few seconds, further enhancing the geometry quality and generating sharp geometric details, as shown in Figure 4.5.

4.3 Experiments

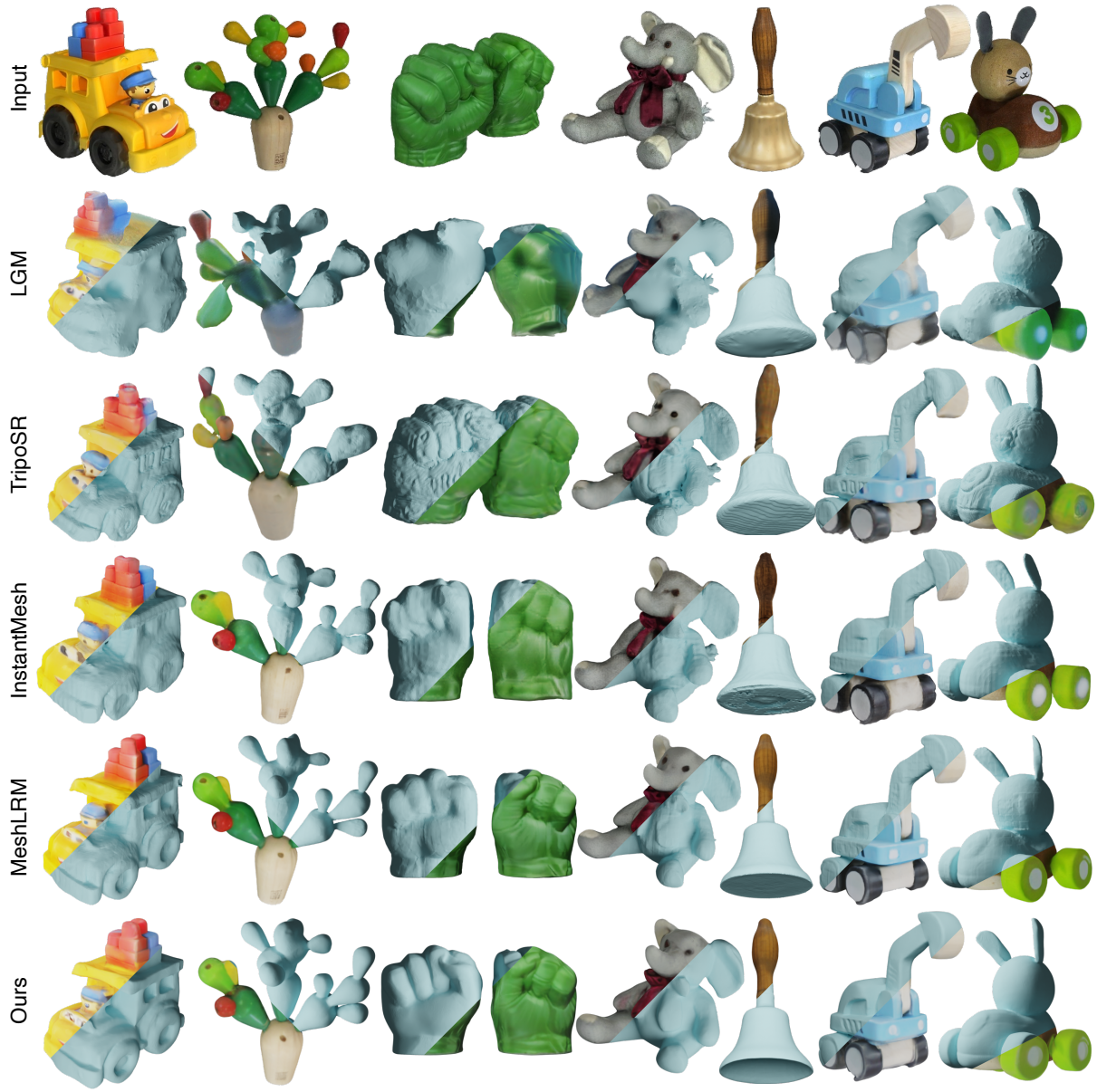


Figure 4.3. Qualitative Comparison of Single-Image-to-3D (GSO dataset). Both the textured and textureless mesh renderings are shown. Please zoom in to examine details and mesh quality.

4.3.1 Implementation Details and Evaluation Settings

Implementation Details We trained MeshFormer on the Objaverse [50] dataset. The total number of network parameters is approximately 648 million. We trained the model using 8 H100

Figure 4.4. Application: Text to 3D. Given a text prompt, a 2D diffusion model first predicts multi-view RGB and normal images, which are then fed to MeshFormer for 3D reconstruction.

GPUs for about one week (350k iterations) with a batch size of 1 per GPU, although we also show that the model can achieve similar results in just two days.

Evaluation Settings We evaluate the methods on two datasets: GSO [59] and OmniObject3D [330]. Both datasets contain real-scanned 3D objects that were not seen during training. For the GSO dataset, we use all 1,030 3D shapes for evaluation. For the OmniObject3D dataset, we randomly sample up to 5 shapes from each category, resulting in 1,038 shapes for evaluation. We utilize both 2D and 3D metrics. For 3D metrics, we use both the F-score and Chamfer distance (CD), calculated between the predicted meshes and ground truth meshes,

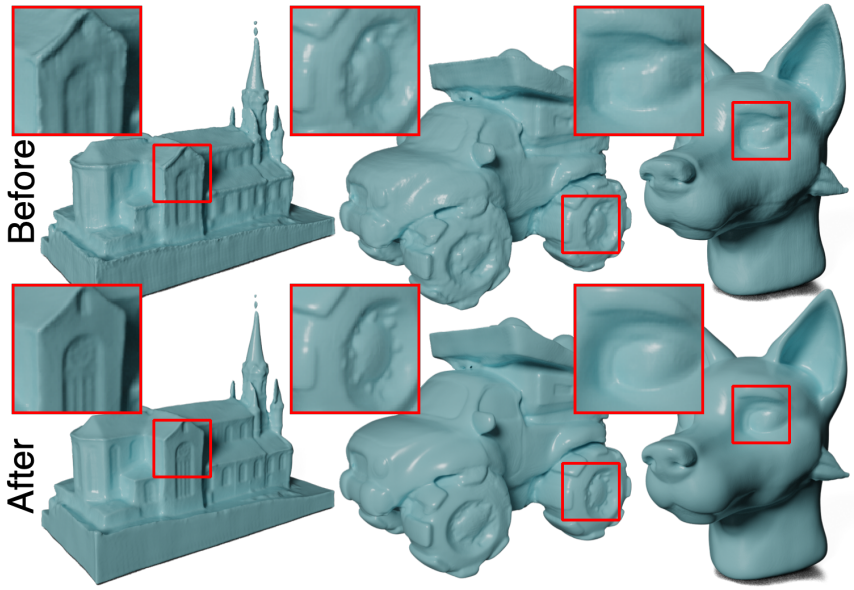


Figure 4.5. Geometry enhancement generates sharper details. Please zoom in to see the details.

following [161, 341]. For 2D metrics, we compute both PSNR and LPIPS for the rendered color images. Since each baseline may use a different coordinate frame for generated results, we carefully align the predicted meshes of all methods to the ground truth meshes before calculating the metrics.

4.3.2 Comparison with Single/Sparse-View to 3D Methods

We compare MeshFormer with recent open-world feed-forward single/sparse-view to 3D methods, including One-2-3-45++ [161], TripoSR [286], CRM [315], LGM [277], InstantMesh [341], and MeshLRM [320]. Many of these methods have been released recently and should be considered concurrent methods. For MeshLRM [320], we contacted the authors for the results. For the other methods, we utilized their official implementations.

Since input settings differ among the baselines, we evaluate all methods in a unified single-view to 3D setting. For the GSO dataset, we utilized the first thumbnail image as the single-view input. For the OmniObject3D dataset, we used a rendered image with a random pose as input. For One-2-3-45++ [161], InstantMesh [341], MeshLRM [320], and our MeshFormer, we first utilized Zero123++ [262] to convert the input single-view image into multi-view images

before 3D reconstruction. Other baselines follow their original settings and take a single-view image directly as input. In addition to the RGB images, our MeshFormer also takes additional multi-view normal images as input, which are also predicted by Zero123++ [262]. **Note that when comparing with baseline methods, we never use ground truth normal images to ensure a fair comparison.**

In Figure 4.3, we showcase qualitative examples. Our MeshFormer produces the most accurate meshes with fine-grained, sharp geometric details. In contrast, baseline methods produce inferior mesh quality. For example, TripoSR directly extracts meshes from the learned NeRF representation, resulting in significant artifacts. While InstantMesh and MeshLRM use mesh representation in their second stage, notable uneven artifacts are still observable upon a zoom-in inspection. Additionally, all baseline methods incorrectly close the surface of the copper bell. We also provide quantitative results in Tab. 4.1. Although our baselines include four methods released just one or two months before the time of submission, our MeshFormer significantly outperforms many of them and achieves the best performance on most metrics across two datasets. For the color LPIPS metric, our performance is very similar to MeshLRM's, despite a perceptual loss being their main training loss term. We also highlight that many of the baselines require over one hundred GPUs for training, whereas our model can be efficiently trained with just 8 GPUs. Please refer to Sec. 4.3.4 for analysis on training efficiency.

4.3.3 Application: Text to 3D

In addition to the single image to 3D, MeshFormer can also be integrated with 2D diffusion models to enable various 3D object generation tasks. For example, we follow the framework proposed by [183] to finetune Stable Diffusion [249] and build a text-to-multi-view model. By integrating this model, along with the normal prediction from Zero123++ [262], with MeshFormer, we can enable the task of text to 3D. Figure 4.4 shows some interesting results, where we convert a single text prompt into a high-quality 3D mesh in just a few seconds.

Table 4.1. Quantitative Results of Single Image to 3D. Evaluated on the 1,030 and 1,038 3D shapes from the GSO [59] and the OmniObject3D [330] datasets, respectively. One-2-3-45++ [161], InstantMesh [341], MeshLRM [320], and our method all take the same multi-view RGB images predicted by Zero123++ [262] as input. CD denotes Chamfer Distance.

Method	GSO [59]				OmniObject3D [330]			
	F-Score ↑	CD ↓	PSNR ↑	LPIPS ↓	F-Score ↑	CD ↓	PSNR ↑	LPIPS ↓
One-2-3-45++ [161]	0.936	0.039	20.97	0.21	0.871	0.054	17.08	0.31
TripoSR [286]	0.896	0.047	19.85	0.26	0.895	0.048	17.68	0.28
CRM [315]	0.886	0.051	19.99	0.27	0.821	0.065	16.01	0.34
LGM [277]	0.776	0.074	18.52	0.35	0.635	0.114	14.75	0.45
InstantMesh [277]	0.934	0.037	20.90	0.22	0.889	0.049	17.61	0.28
MeshLRM [320]	<u>0.956</u>	<u>0.033</u>	<u>21.31</u>	**0.19**	<u>0.910</u>	<u>0.045</u>	<u>18.10</u>	**0.26**
Ours	**0.963**	**0.031**	**21.47**	<u>0.20</u>	**0.914**	**0.043**	**18.14**	<u>0.27</u>

Table 4.2. We compare methods using limited training resources. Evaluated on the GSO [59] dataset.

Method	Training Resources	F-Score ↑	CD ↓	PSNR-C ↑	LPIPS-C ↓	PSNR-N ↑	LPIPS-N ↓
MeshLRM [320]	8×H100 48h	0.925	0.0397	21.09	0.26	21.69	0.22
Ours		0.960	0.0317	21.41	0.20	23.01	0.15

4.3.4 Analysis and Ablation Study

Explicit 3D structure vs. Triplane In Section 4.3.2, we demonstrated that MeshFormer outperforms baseline methods that primarily utilize the triplane representation. Here, we highlight two additional advantages of using the explicit 3D voxel structure: training efficiency and the avoidance of "triplane artifacts". Without leveraging explicit 3D structure, existing triplane-based large reconstruction models require extensive computing resources for training. For example, TripoSR requires 176 A100 GPUs for five days of training. InstantMesh relies on OpenLRM [90], which requires 128 A100 GPUs for three days of training. MeshLRM also utilizes similar resources during training. By utilizing explicit 3D structure and projective bias, our MeshFormer can be trained much more efficiently using only 8 GPUs. To better understand the gap, we trained both MeshLRM and our MeshFormer under very limited training resources, and the results are shown in Table 4.2. When using only 8 GPUs for two days, we found that MeshLRM failed to converge and experienced significant performance degradation compared to

Figure 4.6. The triplane-based method MeshLRM [320] has difficulty capturing words on objects, even when ground truth multi-view RGB images are used as input.

Table 4.3. Ablation Study on the GSO [59] dataset. -C denotes color renderings, and -N denotes normal renderings. CD stands for Chamfer distance. By default, ground truth multi-view images are used to exclude the influence of errors from 2D diffusion models.

	Setting	PSNR-C ↑	LPIPS-C ↓	PSNR-N ↑	LPIPS-N ↓	F-Score ↑	CD ↓
a	w/o normal input	24.82	0.129	24.85	0.107	0.964	0.024
b	w/o SDF supervision	20.72	0.244	20.42	0.257	0.940	0.035
c	w/o transformer layer	26.63	0.101	29.80	0.036	0.992	0.013
d	w/o projection-aware cross-attention	25.48	0.155	29.01	0.045	0.991	0.013
e	w/o geometry enhancement	27.95	0.085	29.10	0.048	0.992	0.012
f	w/ pred normal	26.84	0.096	26.99	0.067	0.987	0.017
g	full	28.15	0.083	29.80	0.036	0.992	0.012

the results shown in Table 4.1, while our MeshFormer had already converged to a decent result, close to the fully-trained version, demonstrating superior training efficiency.

We observe that the triplane typically generates results with axis-aligned artifacts, as shown in Figure 4.3 (5th row, please zoom in). As demonstrated in Figure 4.6, these artifacts also cause difficulties for MeshLRM [320] in capturing the words on objects. These limitations are likely caused by the limited number of triplane tokens (e.g., $32 \times 32 \times 3$), constrained by the global attention, which often leads to artifacts at the boundaries of the triplane patches. In contrast, MeshFormer leverages sparse voxels, supports a higher feature resolution of 256^3, and is free from such artifacts.

Normal Input and SDF supervision As shown in Table 4.3 (a), the performance significantly drops when multi-view input normal maps are removed, indicating that the geometric guidance

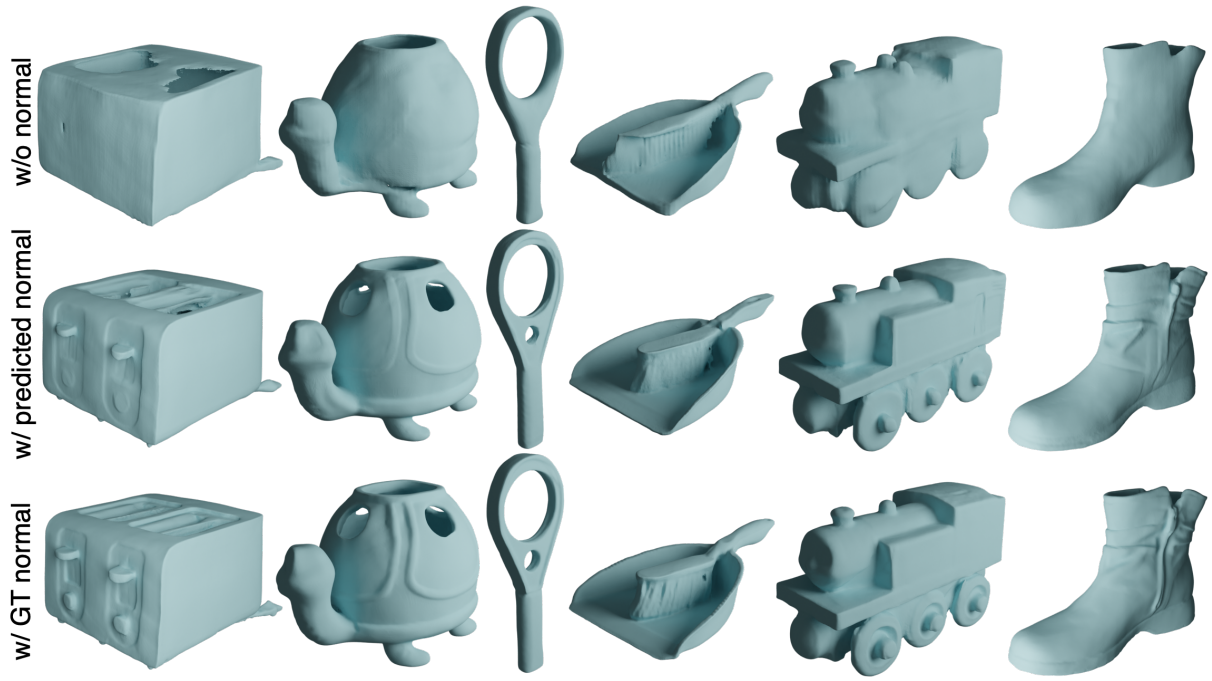


Figure 4.7. Ablation study on input normal maps. Evaluated on the GSO dataset [59]. "w/o normal" indicates that the model is trained with multi-view RGB images only. "w/ predicted normal" indicates that the model is trained with ground truth normal maps but evaluated with predicted normals by Zero123++ [262]. "w/ GT normal" indicates that the model is trained and tested with ground truth normals.

and clues provided by normal images are crucial for facilitating network training, particularly for local geometric details. In (f), we replace ground truth normal maps with normal predictions by Zero123++ [262] and observe a notable performance gap compared to (g). This indicates that although predicted multi-view normal images can be beneficial, existing 2D diffusion models still have room for improvement in generating more accurate results. See Figure 4.7 for qualitative examples. As shown in (b), if we remove the SDF loss after the first epoch and train the network using only surface rendering losses, the geometry learning quickly deteriorates, resulting in poor geometry. This explains why existing methods [141, 320] typically employ complex multi-stage training and use volume rendering to learn a coarse NeRF in the initial stage. By leveraging explicit 3D SDF supervision as strong geometric regularization, we enable a unified single-stage training, using mesh as the only representation.

Projection-Aware Cross-Attention and Transformer Layers We propose to utilize projection-

aware cross-attention to precisely aggregate multi-view projected 2D features for each 3D voxel. In conventional learning-based multi-view stereo (MVS) methods [26, 184], average or max pooling is typically employed for feature aggregation. In Table 4.3 (d), we replace the cross-attention with a simple average pooling and we observe a significant performance drop. This verifies that projection-aware cross-attention provides a more effective way for 3D-2D interaction while simple average pooling may fail to handle the occlusion and visibility issues. In the bottleneck of the UNet, we treat all 3D (sparse) voxels as a sequence of tokens and apply transformer layers to them. As shown in row (c), after removing these layers, we observe a performance drop in metrics related to texture quality. This indicates that texture learning requires more extensive priors and benefits more from the transformer layers.

Geometry Enhancement We propose to learn an additional normal map texture and apply a traditional algorithm as post-processing for geometry enhancement during inference. As shown in Figure 4.5, the geometry enhancement aligns the mesh geometry with the learned normal texture and generates fine-grained sharp details. In some cases (such as the wolf), the meshes output by the network are already good enough, and the difference caused by the enhancement tends to be subtle. Row (e) also quantitatively verifies the effectiveness of the module.

4.4 Summary

We present MeshFormer, an open-world sparse-view reconstruction model that leverages explicit 3D native structure, supervision signals, and input guidance. MeshFormer can be conveniently trained in a unified single-stage manner and efficiently with just 8 GPUs. It generates high-quality meshes with fine-grained geometric details and outperforms baselines trained with over one hundred GPUs.

MeshFormer relies on 2D models to generate multi-view RGB and normal images from a single input image or text prompt. However, existing models still have limited capabilities to generate consistent multi-view images, which can cause a performance drop. Strategies to

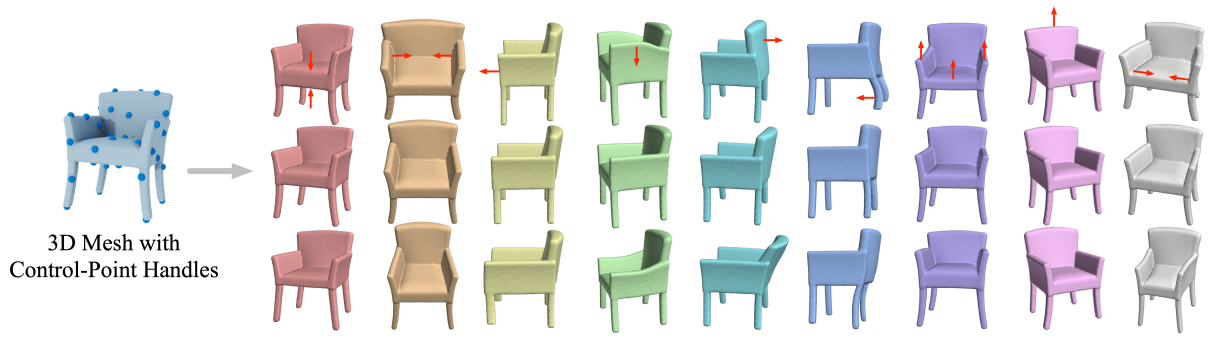


Figure 4.8. Learned meta-handles for a single chair. Each column indicates a meta-handle and shows three deformations along the direction of that meta-handle, with red arrows highlighting the deformed region. Our method learns intuitive and disentangled meta-handles in an unsupervised fashion, which factorize all the plausible deformations for the shape.

improve model robustness against such imperfect predictions are worth further exploration, and we leave this as future work.

4.5 Other Related Projects on Leveraging 3D Priors for Reconstruction and Generation

In MeshFormer, we find that by incorporating various types of 3D-native priors into the model design, we can significantly improve both the mesh quality and the training efficiency of the feed-forward sparse-view reconstruction model.

Additionally, I have two related projects that also leverage explicit 3D priors to benefit learning-based 3D generation and reconstruction. One project aims to learn all plausible deformations of a given input shape, enabling an automated approach for the self-proliferation of 3D meshes. The other project focuses on the long-standing problem of meshing point clouds and introduces a novel learning-based algorithm to learn priors from data that aid in reconstructing ambiguous structures.

4.5.1 Learning Deformation Meta-Handles of 3D Meshes

3D Meshes can store sharp edges and smooth surfaces compactly. However, Learning to generate 3D meshes is much more challenging than 2D images due to the irregularity of

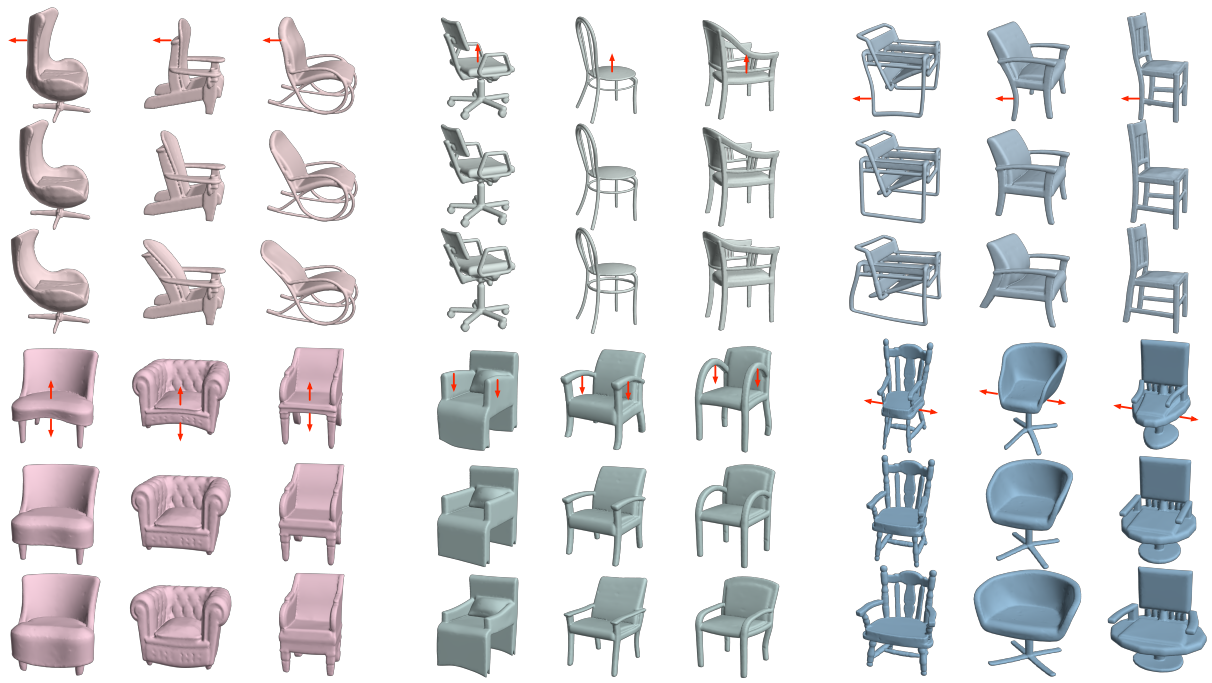


Figure 4.9. Learned meta-handles across different shapes. The figure includes six meta-handles, and each color indicates a distinct one. For each meta-handle, the figure demonstrates the corresponding deformations on three different shapes, with the red arrows highlighting the deformation direction. The meta-handles are consistent across various shapes.

mesh data structures and the difficulty in designing loss functions to measure geometrical and topological properties. For such reasons, to create new meshes, instead of generating a mesh from scratch, recent work assumes that the connectivity structure of geometries is known so that the creation space is restricted to changing the geometry without altering the structure. For example, [273, 274] create new shapes by *deformations* of one template mesh. They, however, limit the scope of the shape generation to possible variants of the template mesh. We thus propose a 3D *conditional* generative model that can take any existing mesh as input and produce its plausible variants. Our approach integrates a *target-driven* fitting component and a conditional generative model. At test time, it allows both deforming the input shape to fit the given target shape and exploring plausible variants of the input shape without a target.

Our main design goals are two-fold: improving the *plausibility* of the output shapes and enhancing the *interpretability* of the learned latent spaces. To achieve the goals, the key

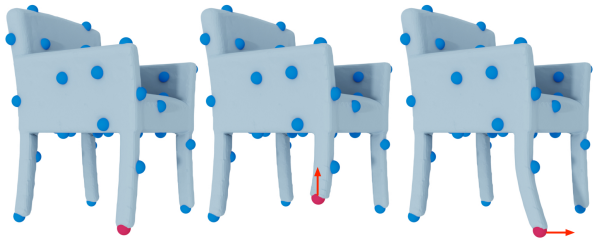


Figure 4.10. Two deformations resulted from moving the red control point along the arrow directions.

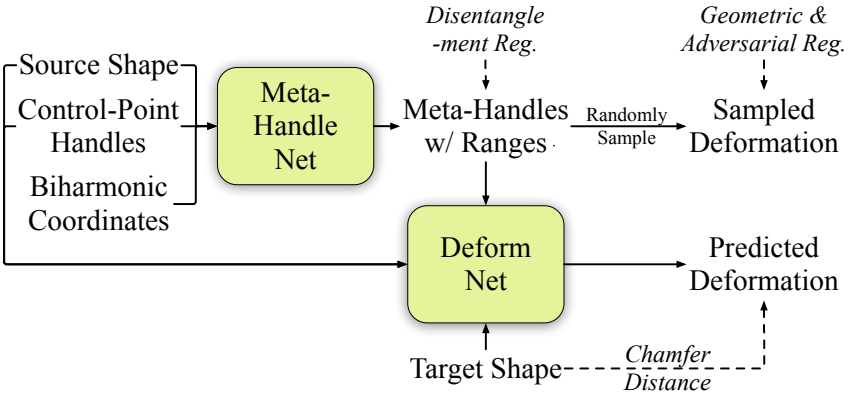


Figure 4.11. Method overview of DeepMetaHandles. We learn the meta-handles in an unsupervised fashion.

is to choose a suitable parameterization of deformations. One option is to follow the recent target-driven deformation network [79, 272, 307, 366], which parameterizes the deformation as new positions of all the mesh vertices. However, such a large degree of freedom often results in the loss of fine-grained geometric details and tends to cause undesirable distortions. Instead of following the above works, we leverage a classical idea in computational geometry, named *deformation handles*, to parameterize smooth deformations with a low degree of freedom. Specifically, we propose to take a small set of *control points* as deformation handles and utilize a deformation function defined on the control points and their *biharmonic coordinates* [311]. Please refer to Figure 4.10 for the illustration.

Not all the translations of the control points lead to plausible deformations. Based on the control-point handles, we aim to learn a low-dimensional deformation subspace for each shape, and we expect the structure of this subspace to exhibit *interpretability*. In contrast to

typical generative models, where shape variations are embedded into a latent space implicitly, our method explicitly factorizes all the plausible deformations of a shape with a small number of interpretable deformation functions. Specifically, for each axis of our input-dependent latent space, we assign a deformation function defined with the given set of control points and offset vectors on them so that each axis corresponds to an intuitive deformation direction. Since each axis is explicitly linked to multiple control-point handles, we thus call them *meta-handles*. We enforce the network to learn *disentangled* meta-handles, in the sense that a meta-handle should not only leverage the correlations of the control-point handles, but also correspond to a group of parts that tend to deform altogether according to the dataset. We hope that the disentangled meta-handles allow us to deform each part group independently in downstream applications.

Beyond choosing the parameterization of deformations, we have to overcome the challenge of examining the plausibility. In the popular adversarial learning framework, a straightforward approach would be converting the output mesh to voxels or point clouds and exploiting voxel or point cloud based discriminators. The conversions, however, may discard some important geometric details. In our method, we instead project the shapes into a 2D space with a differentiable *soft rasterizer* [171] and employ a 2D discriminator. We found that this architecture can be trained more robustly, and it captures local details of plausible shapes.

Our deformation-based conditional generative model, named **DeepMetaHandles**, takes random pairs of source and target shapes as input during training. For the source shape, the control points are sampled from its mesh vertices by farthest point sampling, and the biharmonic cooridnates [311] for control-point handles are pre-computed. As shown in Figure 4.11, our network consists of two main modules: MetaHandleNet and DeformNet. The MetaHandleNet first predicts a set of meta-handles for the source shape, where each meta-handle is represented as a combination of control-point offsets. A deformation range is also predicted for each meta-handle, describing the scope of plausible deformations along that direction. The learned meta-handles, together with the corresponding ranges, define a deformation subspace for the source shape. Then, DeformNet predicts coefficients multiplied to the meta-handles, within the predicted

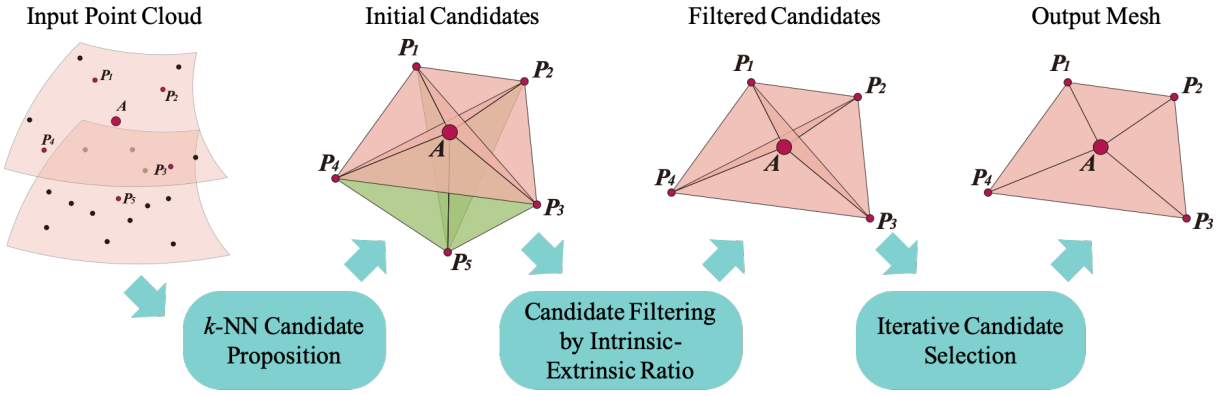


Figure 4.12. The general pipeline of our remeshing algorithms from a local view. The candidate triangles close to the underlying surfaces are marked in pink.

ranges, so that the source shape deformed with the coefficients can match the target shape. To ensure the plausibility of variations within the learned subspace, we then randomly sample coefficients within the predicted ranges and apply both geometric and adversarial regularizations to the corresponding deformations.

Figure 4.8 shows examples of the learned meta-handles, which interestingly resemble natural deformations of *semantic* parts, such as lifting the armrests or bending the back of a chair. Our experiments also show that the learned meta-handles are consistent across various shapes and well disentangle the shape variation space (see Figure 4.9). Finally, we compare our approach with other target-driven deformation techniques [79, 104, 307, 366] and demonstrate that our method produces superior fitting results.

4.5.2 Meshing Point Clouds with Predicted Intrinsic-Extrinsic Ratio Guidance

Reconstructing high-quality 3D meshes from point clouds thus has been studied for quite a long time and serves as a prerequisite for numerous real-world applications, including autonomous driving, augmented reality, and robotics. Despite its long history, the mesh reconstruction problem remains unresolved. Traditional methods [16, 125, 185] typically reconstruct the mesh either by explicitly connecting the points or implicitly approximating the surface,

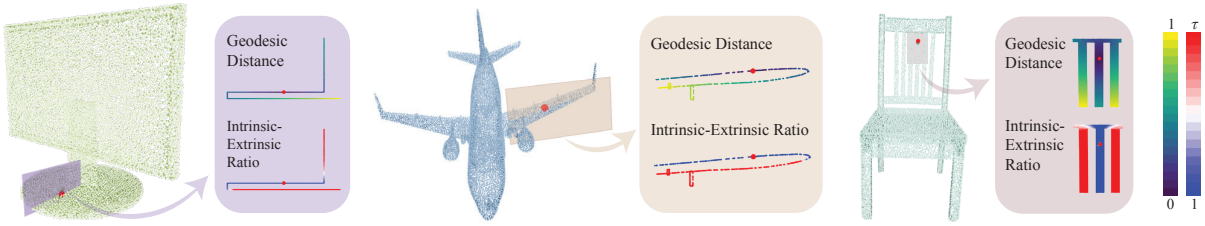


Figure 4.13. In each example, we sample a slice from the input point cloud, and demonstrate the geodesic distance and the intrinsic-extrinsic ratio (IER) of each point to the key point (marked in red).

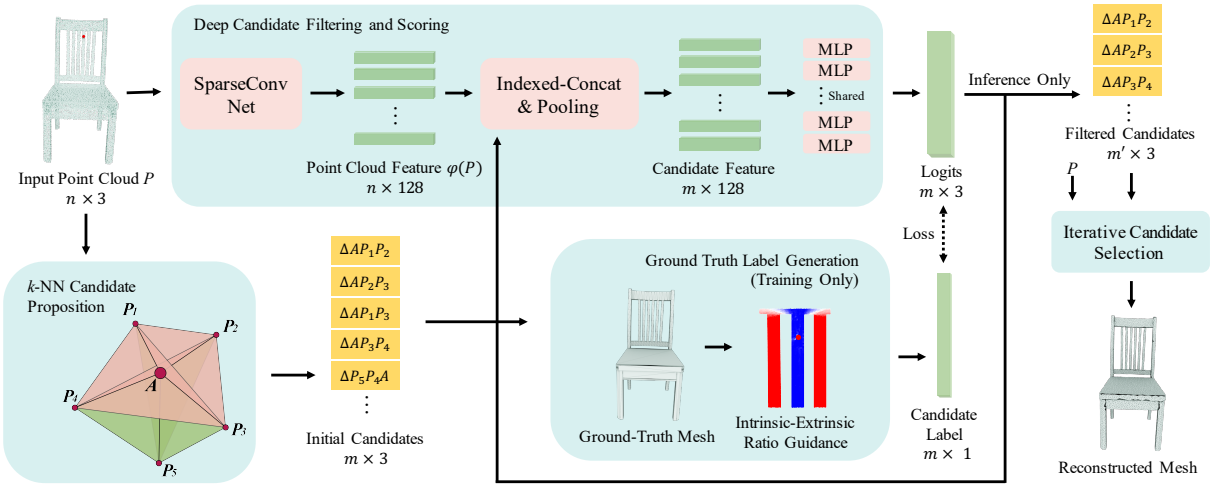


Figure 4.14. The figure shows the full pipeline of our reconstruction algorithm. Given a point cloud as input, we first propose a set of candidate triangles. During training, the network is trained to classify the candidate triangles with intrinsic-extrinsic ratio supervision from the ground truths. During inference, the predicted label is used to filter out and score the candidate triangles, which are then merged into the output mesh by our iterative candidate selection algorithm.

both of which resort to local geometric hints. Without reasoning about the shape, traditional methods may be hard to handle the ambiguous structures when the resolution of the input point cloud is limited. For example, the ambiguous structures may include thin structures consisting of two very close surfaces, independent but spatially adjacent parts, and corners. Traditional methods tend to produce distortion or connect independent parts incorrectly when facing these structures. However, the reconstruction of these fine-grained structures may be essential for many downstream applications such as robotics grasping which needs an accurate understanding of part-level mobility. Moreover, traditional methods are typically sensitive to hyper-parameters.

For most of these methods, a dedicated parameter-tuning is required for each input, making batch processing of point clouds impractical.

With the rapid development of 3D deep learning and the availability of large-scale 3D datasets, people tend to learn geometric or semantic priors from data. Unlike 2D images and 3D voxels, polygon meshes is an irregular geometric representation, which prevents it from being generated by the neural network directly. However, there are still lots of attempts to explore the neural-network-compatible representations for mesh generation, including template meshes with deformation [68, 76, 77, 123, 171, 220, 300, 322], 2D squares with folding [56, 78, 361], primitives with assembly [33, 260, 288], implicit field function [35, 71, 199, 222], and meshlets with optimization [12, 324]. Existing learning-based methods typically follow the "encoder-decoder" paradigm. The limited capability of the network prevents existing methods from generating fine-grained structures and details. Also, since most existing methods learn the priors at the object level, they tend to memorize the overall shapes and typically cannot generalize to unseen categories.

To this end, we propose a novel method that reconstructs meshes from point clouds by leveraging the intermediate representation of triangle faces [166]. Unlike existing methods, our method fully utilizes the input point clouds, which are on the ground truth surface in most cases, and then estimate the local connectivity with the help of learned guidance. More specifically, as shown in Figures 4.14 and 4.12, we first propose a set of candidate triangle faces, which could be the elements of the reconstructed mesh, by constructing a k-nearest neighbor (k-NN) graph on the input point cloud. We then utilize the neural network to filter out the incorrect candidates and provide cues for sorting the remaining candidates. We find that the ratio of geodesic distance (intrinsic metric) and Euclidean distance (extrinsic metric) between two vertices may provide strong cues for inferring the connectivity and can naturally serve as the supervision for the candidate classification task (see Figure 4.13). Since there are multiple ways to triangulate a surface, we only filter out those candidates that should never appear in the reconstructed mesh, such as the candidates linking two independent parts. A greedy post-processing algorithm is then

used to sort all the remaining candidates and merge them into the final mesh.

We demonstrate that our algorithm can preserve fine-grained details and handle ambiguous structures with the help of learned intrinsic-extrinsic guidance, as shown in Figure 4.15. Since our method reconstructs meshes by estimating local connectivity, which relies mainly on the local geometric information, it can well generalize to unseen categories. In experiments on the ShapeNet dataset, our method outperforms both the existing traditional methods and learning-based mesh generative methods with regard to all commonly used metrics, including the F-score, Chamfer distance, and normal consistency. We also provide extensive ablation studies on different sampling densities, sampling strategies, noisy levels, and real scans to demonstrate our generalizability and robustness.

Chapter 4 has been partially submitted for peer review as the paper titled "MeshFormer: High-Quality Mesh Generation with a 3D-Guided Reconstruction Model", authored by Minghua Liu, Chong Zeng, Xinyue Wei, Ruoxi Shi, Linghao Chen, Chao Xu, Mengqi Zhang, Zhaoning Wang, Xiaoshuai Zhang, Isabella Liu, Hongzhi Wu, and Hao Su. The dissertation author was the primary investigator and lead author of this paper.

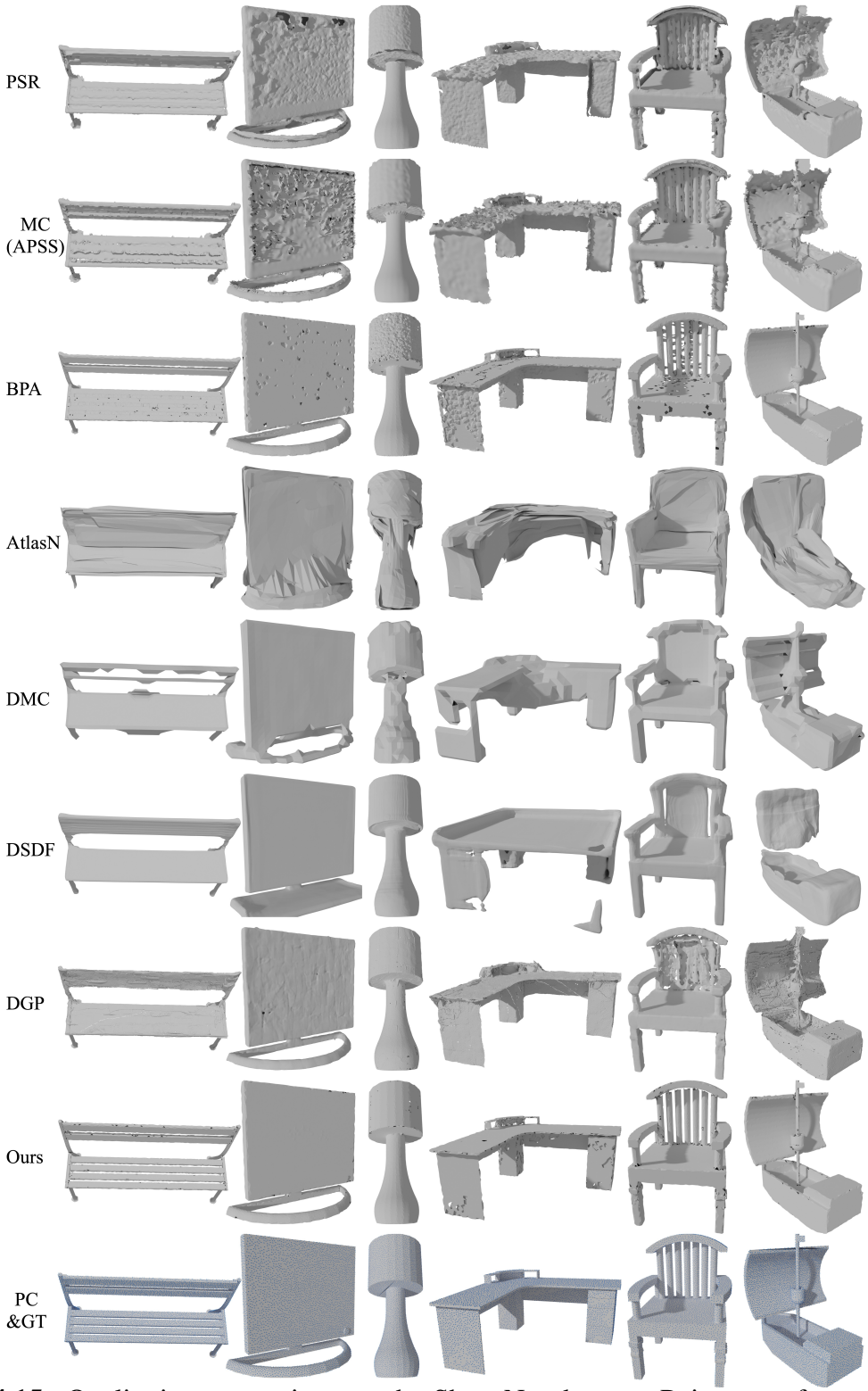


Figure 4.15. Qualitative comparison on the ShapeNet dataset. Poisson surface reconstruction [125, 126], marching cube (APSS) [80, 185], ball-pivoting algorithm [16], AtlasNet [78], Deep Marching Cubes [149], DeepSDF [222], Deep Geometric Prior [324], our method, and ground-truth meshes with input point clouds are shown from top to bottom.

Chapter 5

Open-World 3D Understanding: Multi-Modal Representation Learning

In addition to open-world 3D object generation, open-world 3D object-level understanding also plays a critical role in various applications, such as augmented/virtual reality, autonomous driving, and robotics. A straightforward approach involves projecting 3D data into the 2D domain through rendering, followed by the use of large-scale pre-trained 2D vision models, such as CLIP, to analyze the 2D images. This enables zero-shot 3D shape classification [383, 396]. However, these methods are hindered by occlusion, information loss during projection, and unnecessary latency caused by point cloud rendering and multiple CLIP inferences.

To overcome the limitations caused by projection, it is necessary to train a 3D-native model by distilling knowledge from pretrained 2D models. However, training a 3D-native model requires a set of 3D shapes, and the amount of knowledge that can be distilled is determined by the size of the 3D dataset. For example, ULIP [350] aims to learn a joint representation space between language, 2D images, and 3D shapes, but uses a small-scale 3D dataset ShapeNetCore [22] for knowledge distillation. Specifically, ULIP fixes the 2D CLIP text and image encoders and trains a dedicated 3D-native point cloud encoder to extract 3D shape representations. The 3D encoder strives to align the 3D shape embedding space with the CLIP image and language embedding spaces by utilizing contrastive learning across all three modalities. However, since ULIP is only trained on 52K shapes of 55 object categories, it still struggles with out-of-distribution shape

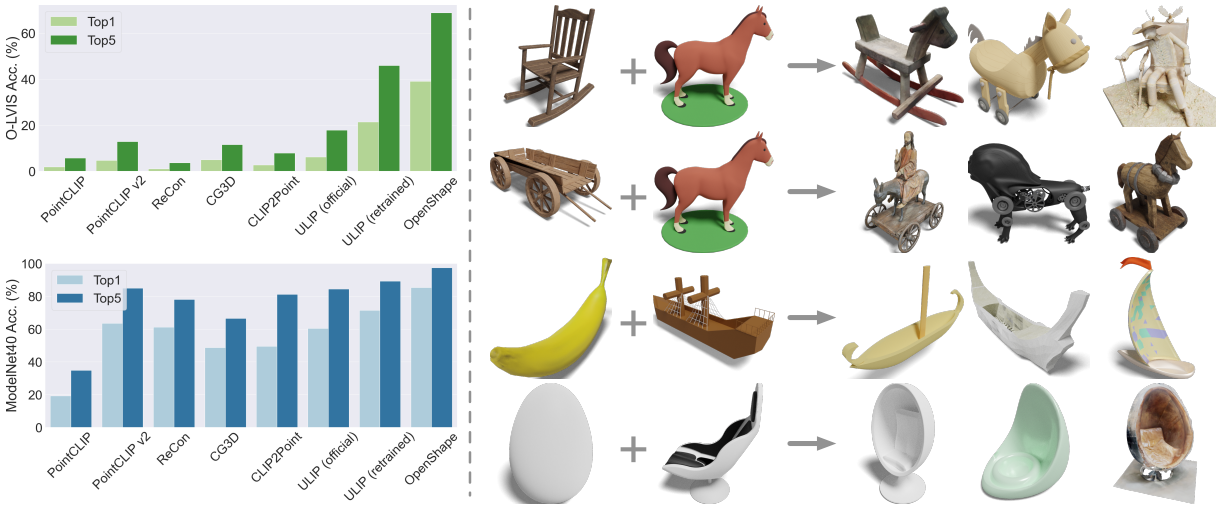


Figure 5.1. Left: Zero-shot shape classification on the Objaverse-LVIS (1,156 categories) and ModelNet40 datasets. OpenShape outperforms previous methods by a large margin. We exclude shapes in Objaverse-LVIS during training, and we also retrain ULIP [350] on our ensembled training shapes for fair comparison. **Right**: Our shape representations encode a broad range of semantic and visual concepts. We input two 3D shapes and use their shape embeddings to retrieve the top three shapes whose embeddings are simultaneously closest to both inputs. See Section. 5.3.4 for more details.

categories and fails to demonstrate an impressive open-world understanding of 3D shapes.

In this chapter, we propose a novel method called OpenShape, which follows a similar paradigm as ULIP but aims to achieve a more generalized and scalable joint representation space encompassing language, 2D images, and 3D shapes. Our focus mainly lies on scaling up representation learning and addressing corresponding challenges. In OpenShape, we emphasize four key factors during the training process: (a) data scale: we significantly increase the scale of 3D training data by combining four public 3D shape datasets, resulting in 876k 3D shapes covering much more diverse categories; (b) text quality: the 3D shapes from our main dataset, Objaverse [50], is dominated with inaccurate or uninformative text descriptions. Given the data scale, we propose three strategies to automatically filter and enrich the text descriptions; (c) 3D backbone scaling: since most existing 3D backbones target small datasets, we find that it's important but non-trivial to scale up the 3D backbones; and (d) data resampling: since the ensembled dataset is highly unbalanced, we utilize hard negative mining to improve the model's

discriminative ability.

We first evaluate OpenShape on the zero-shot 3D shape classification task. As shown in Figure 5.1, OpenShape outperforms previous zero-shot approaches on the ModelNet40 dataset by at least 20%. Moreover, OpenShape excels at handling long-tail categories. On the challenging Objaverse-LVIS dataset, which contains 1,156 categories, OpenShape achieves a 46.8% accuracy, significantly surpassing previous methods. Notably, this performance gap remains even when ULIP is retrained on our ensembled datasets, highlighting the superiority of our text enrichment and training strategies. Besides zero-shot classification, we present demos that showcase the wide range of visual and semantic concepts learned by OpenShape. For example, in Figure 5.1-right, we take two 3D shapes as input and use their OpenShape embeddings to retrieve the top three shapes whose embeddings are simultaneously closest to both inputs from our ensembled dataset. The retrieved shapes exhibit an interesting combination of the semantic and geometric elements from both input shapes. Furthermore, since we align our 3D shape embedding space with the CLIP language and image embedding space, we demonstrate that OpenShape embeddings can be easily integrated with other CLIP-based models to perform cross-modality tasks such as point cloud captioning and point cloud-conditioned image generation.

5.1 Related Work

5.1.1 CLIP for 3D Learning

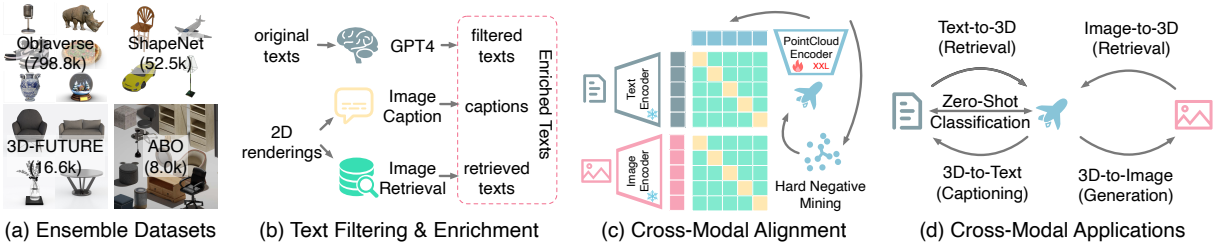
Image-language models like CLIP have achieved remarkable performance through large-scale image-text pretraining [5, 115, 144, 147, 234, 236, 249, 377]. As these models excel at capturing rich visual concepts and possess impressive zero-shot capabilities, they have been applied to various 3D vision tasks. For instance, numerous recent works utilize CLIP to facilitate zero-shot text-to-3D generation [9, 20, 96, 110, 114, 129, 139, 179, 201, 252, 342, 374], typically through CLIP-guided per-scene optimization. From a recognition perspective, some works focus on scene-level representation, aiming to leverage CLIP priors for zero-shot 3D segmentation

or detection in both indoor [58, 85, 105, 113, 128, 188, 224, 248, 358, 374, 379] and outdoor scenes [30, 94]. Meanwhile, another line of work focuses on shape-level understanding, targeting zero-shot shape classification [91, 230, 350, 383, 396] and part segmentation [1, 168]. There are two primary working paradigms for these methods. The first [106, 383, 396] involves using images as a medium representation, projecting 3D point clouds into 2D and employing 2D CLIP for inference. However, these methods typically suffer from occlusion and information loss during projection, along with unnecessary latency due to point cloud rendering and multiple 2D CLIP inferences. The second paradigm involves training a 3D-native encoder attempting to distill or fuse CLIP features into 3D representations. Our paper follows this paradigm.

5.1.2 3D Shape Representation Learning

Various works have studied self-supervised pretraining for point clouds by designing pretext tasks [3, 60, 226, 258, 282] such as self-reconstruction [4, 53, 238, 296], masked auto-encoding [93, 221, 372], distortion reconstruction [198, 253, 269], normal estimation [238], and contrastive learning [251, 338, 388]. These tasks enhance models' shape representations and improve their performance on downstream applications, although they do not involve multimodal semantic alignments during pretraining.

Recently, some works [91, 230, 350], exemplified by ULIP [350], have explored learning multimodal joint representations for 3D shapes. They train 3D-native shape encoders by aligning 3D shape embeddings with CLIP's language and/or image embeddings through multimodal contrastive learning. Works like ReCon [230] further combines cross-modal contrastive learning with masked auto-encoding for added enhancement. While these methods allow for zero-shot 3D classification through the computation of 3D-text similarity, the amount of distilled knowledge and their model capability are heavily limited by the small-scale training datasets used. Our work follows this paradigm but aims to learn more generalizable and scalable representations to enable open-world 3D shape understanding.



(a) Ensemble Datasets (b) Text Filtering & Enrichment (c) Cross-Modal Alignment (d) Cross-Modal Applications

Figure 5.2. (a) We ensemble four public 3D shape datasets, resulting in 876k shapes that encompass diverse categories and concepts. (b) We propose three strategies to automatically filter and enrich the noisy texts in the original datasets. (c) We train a 3D point cloud encoder to align the 3D shape embedding space with the CLIP's text and image embedding spaces. We perform cross-modal contrastive learning with scaled 3D backbones and hard negative mining. (d) OpenShape embeddings can be easily integrated with other CLIP-based models, enabling various cross-modality tasks.

5.2 Proposed Method: OpenShape

We propose a novel method, *OpenShape*, for learning generalizable and scalable multi-modal joint representation between language, 2D images, and 3D shapes, as shown in Figure 5.2. We first introduce the multi-modal contrastive learning framework we used for aligning representations of three modalities in Section 5.2.1. We then elaborate how we create our training sets and enrich our text data in Sections 5.2.2 and 5.2.3. In Section 5.2.4, we present how we scale up our 3D backbone models. Finally, we propose a hard negative mining strategy to enhance contrastive learning in Section 5.2.5.

5.2.1 Multi-Modal Representation Alignment

We aim to learn 3D shape representations that are aligned with pretrained CLIP embedding spaces of language and image. As shown in Figure 5.2 (c), we train a 3D native encoder f^P that takes a 3D point cloud as input and extracts 3D shape feature. Following previous works [91, 230, 350], such as ULIP [350], we utilize multi-modal contrastive learning for representation alignment. Since CLIP is pretrained on a much larger scale data, we freeze both its text encoder f^T and its image encoder f^I during feature alignment to preserve CLIP's feature priors and avoid model collapse. Specifically, given a sampled batch of triplets $\{(P_i, T_i, I_i)\}$,

where P_i denotes a point cloud of a 3D shape, T_i and I_i denote corresponding text and image, the contrastive loss is calculated as:

$$-\frac{1}{4n}\sum_i\left(\log\frac{\exp(h_i^P\cdot h_i^T/\tau)}{\sum_j\exp(h_i^P\cdot h_j^T/\tau)}+\log\frac{\exp(h_i^T\cdot h_i^P/\tau)}{\sum_j\exp(h_i^T\cdot h_j^P/\tau)}+\log\frac{\exp(h_i^P\cdot h_i^I/\tau)}{\sum_j\exp(h_i^P\cdot h_j^I/\tau)}+\log\frac{\exp(h_i^I\cdot h_i^P/\tau)}{\sum_j\exp(h_i^I\cdot h_j^P/\tau)}\right) \quad (5.1)$$

where n is the number of shapes in a batch; τ is a learnable temperature; $h_i^P = f^P(P_i)/|f^P(P_i)|$, $h_i^T = g^T(f^T(T_i))/|g^T(f^T(T_i))|$, and $h_i^I = g^I(f^I(I_i))/|g^I(f^I(I_i))|$ denote normalized projected features of P_i, T_i, and I_i, where g^T and g^I are two learnable linear projections. Since f^T and f^I are frozen, we extract all $f^T(T_i)$ and $f^I(I_i)$ before training and cache them for acceleration. In most of our experiments, we utilize OpenCLIP ViT-bigG-14 [109] as the pretrained CLIP model.

5.2.2 Ensembling 3D Datasets

Since the scale and diversity of training triplets play a crucial role in learning scalable shape representations, we ensemble four currently-largest public 3D datasets for training as shown in Figure 5.2 (a), resulting in 876k training shapes. Among these four datasets, ShapeNet-Core [22], 3D-FUTURE [63] and ABO [46] are three popular datasets used by prior works. They contain human-verified high-quality 3D shapes, but only cover a limited number of shapes and dozens of categories. The Objaverse [50] dataset is a more recent dataset, containing many more 3D shapes and covering significantly more diverse categories. However, shapes in Objaverse are mainly uploaded by web users and not verified by experts, and thus have uneven quality and exhibit highly unbalanced distributions, necessitating further processing.

To create triplets for training, for each shape, we sample 10,000 points from the mesh surface and interpolate the point colors according to the mesh textures. We also render 12 color images from the preset camera poses that uniformly cover the whole shape. For datasets providing thumbnails, we include them as part of image candidates, since they typically capture the shape from a better camera view. For the Objaverse dataset, we use the model name as the raw text for each shape. For other datasets, we utilize provided metadata to create raw texts. During each pretraining iteration, we randomly sample one rendered image or thumbnail for each shape, and apply standard augmentation to the point clouds [350].

Figure 5.3. Text Filtering & Enrichment Examples. In each example, the left section features the thumbnail, model name, and GPT-4 filtering results. The upper right section shows image captions from two captioning models, while the lower right section displays retrieved images and their corresponding texts.

5.2.3 Text Filtering and Enrichment

We find that only applying contrastive learning between 3D shapes and 2D images is insufficient to fuel zero-shot 3D classification, even when training on large-scale datasets. We conjecture that this is caused by the inherent domain gap in CLIP's language and image embedding spaces, which is also observed by previous studies [148, 289]. Consequently, 3D-text alignment is not guaranteed even if we obtain good 3D-image alignments via contrastive learning. Therefore, we need to explicitly align 3D shapes with text. Along this process, to facilitate better 3D-text alignment, we introduce 3 techniques to improve the text quality: filtering, captioning, and image retrieval, as shown in Figure 5.2 (b).

Filtering. As shown in Figure 5.3, the 3D shapes from our main dataset, Objaverse, is dominated with noisy text descriptions (“names”) uploaded by web users. Many of the problematic texts can be identified from the text itself without seeing the corresponding 3D shape. We thus leverage a powerful large language model, GPT-4 [217], to filter out inaccurate or uninformative text descriptions. We find that GPT-4 excels at recognizing irrelevant contents, such as timestamps, pure model numbers, incomprehensible descriptions, random filenames (e.g., new project), and

random characters. Through GPT-4, we filter out about 30% of raw user texts. Note that we only filter the texts, and still keep all shapes for training.

Captioning. We utilize BLIP [142] and the Azure cognition services to caption the 2D thumbnails (if present, or images rendered from a fixed frontal view) of the 3D models, obtaining two texts for each shape. As shown in Figure 5.3, the captioning models can usually produce meaningful and descriptive captions that either enhance user-uploaded texts or replace low-quality ones. We also notice that the two caption models complement each other, leading to better performance.

Image Retrieval. In addition to image captioning, we also perform image retrieval to obtain additional descriptions of 3D models. We retrieve k-NN images of shape renderings from the LAION-5B dataset [256] using the CLIP ViT-L retrieval index [13]. We then take the captions of the k-NN images as the retrieved texts for our 3D models. Compared with captioning model generations, retrieved texts cover a wider range of text styles. They can also include more fine-grained semantics than both the user texts and the generated captions (e.g., "Labrador" in Figure 5.3).

In each iteration of pretraining, for each shape, we first randomly sample a text source category among the raw text (if unfiltered), the captions, and the retrieved texts. We then select a text candidate from the selected category. We also apply the template-based prompt engineering technique used in ULIP [350] to both training texts and test-time category names. Specifically, we extend a word or a phrase to a collection of templated simple sentences and take their average embedding.

5.2.4 Scaling Up 3D Point Cloud Backbones

Previous works on 3D point cloud learning have primarily focused on smaller-scale datasets like ShapeNet. These techniques may not be directly applicable to our larger-scale ensembled dataset and need to be scaled up accordingly. We find that different 3D backbones may exhibit distinct behavior and scalability when trained on datasets with varying sizes. Specif-

Table 5.1. Comparison of different 3D backbones **before scaling up their parameters**. Models are trained on ShapeNet [22] or our ensembled dataset excluding Objaverse-LVIS [50]. Zero-shot classification performance are evaluated on ModelNet40 [333] and Objaverse-LVIS [50].

Model	#Parameters.	Train on ShapeNet [22]		Train on Ensemble-no-LVIS	
		ModelNet40	Objaverse-LVIS	ModelNet40	Objaverse-LVIS
PointNet [228]	1.3M	67.0	9.3	74.9	24.4
DGCNN [312]	2.3M	67.8	9.0	74.2	24.8
PointMLP [191]	9.3M	73.5	12.9	82.9	36.6
PointNeXt [231]	2.8M	72.6	12.2	81.6	33.8
PointBERT [372]	5.1M	70.3	10.8	84.5	37.0
SparseConv [42]	5.3M	70.7	10.6	78.8	31.7
std. dev.		2.3	1.4	3.9	5.1

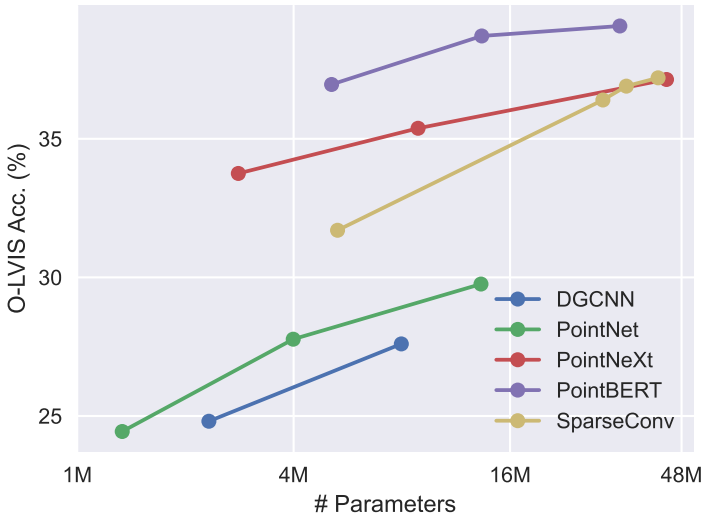


Figure 5.4. Zero-shot classification accuracy on Objaverse-LVIS [50] when scaling up the parameters of different models.

ically, we compare six popular backbones trained on ShapeNet or our ensembled dataset by evaluating their zero-shot classification performance on ModelNet40 [333] and Objaverse-LVIS datasets (for now, these backbones are trained with their original configurations and without scaling up model sizes). **Objaverse-LVIS** is a subset of Objaverse dataset with human-verified category labels. With 1,156 categories, it serves as a suitable dataset for evaluating zero-shot long-tail classification, and we exclude all shapes of Objaverse-LVIS from this experiment. Results are shown in Table 5.1. We find that when trained on ShapeNet, all backbones share similar performances. However, when trained on our ensembled dataset, the performance gap

between backbones increases significantly. This suggests that while the original versions of these backbones share a similar number of parameters, some may have been saturated when trained on small datasets, while others do not.

We also explore the performance and scalability of these backbones when scaling up the model sizes and training on our ensembled dataset. As shown in Figure 5.4, we observe that all 3D backbones benefit significantly from model scaling. However, traditional backbones without a shrinking hierarchical structure, such as DGCNN and PointNet, require operating completely on dense points or modeling the relationships (e.g., through kNN) between dense points. As a result, they become more time-consuming and memory-intensive when scaled up compared to more modern backbones. We therefore select PointBERT [372] (Transformer-based) and SparseConv [42] (convolution-based) as our 3D backbones for the remaining experiments, as they exhibit strong performance and scalability.

5.2.5 Hard Negative Mining

Our ensembled dataset exhibits a high degree of class imbalance. Certain common categories, such as building, may occupy tens of thousands of shapes, while many other categories, such as walrus and wallet, are underrepresented with only a few dozen or even fewer shapes. Consequently, when randomly constructing batches, it is unlikely that shapes from two confusing categories (e.g., apples and cherries) will be contrasted within the same batch. Inspired by some previous works [122, 246], we propose an offline hard negative mining strategy for improving the training efficiency and performance. Specifically, in the first round of training, we train our model with random batches until it is about to converge. We then compute the kNN for each shape in the learned 3D embedding space. In the second round of training, for each iteration, we randomly select s seed shapes and then obtain m neighbors from the kNN results of each seed shape, resulting $s \times m$ shapes per batch. In this way, confusing pairs are more likely to be selected in a single batch. However, this may also introduce false negative pairs (e.g., two apples) into contrastive learning. To mitigate this issue, we leverage image and text embeddings

to filter out pairs sharing similar texts when calculating the contrastive loss. Specifically, for two shapes i and j selected from the same seed shape, if $h_j^T \cdot h_i^I + \delta > h_i^T \cdot h_i^I$, where h^T and h^I are text and image embeddings, and δ is a small threshold, we believe that the text embeddings of i and j are very close to each other, and we remove j from i's negative examples when calculating contrastive loss. By employing this strategy to construct batches, we observe faster and better model learning.

5.3 Experiments

5.3.1 Zero-Shot Shape Classification

We evaluate the zero-shot classification performances of our models on three benchmarks: the traditional ModelNet40 [333] and ScanObjectNN [290], as well as a new benchmark, Objaverse-LVIS [50]. ModelNet40 and ScanObjacetNN consist of 40 and 15 common categories, respectively. Objaverse-LVIS is an annotated subset of Objaverse [50] and comprises 46,832 shapes among 1,156 LVIS [83] categories. With a much larger base of classes than other benchmarks, Objaverse-LVIS presents a challenging long-tailed distribution, making it a better reflection on models' performance in open-world scenarios. We compare OpenShape with existing zero-shot approaches, including PointCLIP [383], PointCLIPv2 [396], ReCon [230], CG3D [91], CLIP2Point [106], and ULIP [350]. Among them, PointCLIP [383] and PointCLIPv2 [396] project point clouds into 2D images and directly utilize 2D CLIP for inference, while other methods leverage the CLIP embedding spaces for alignment and require 3D shapes for training. We report results on these baselines using their released checkpoints. To better analyze the source of our performance gains, we also retrain the baseline ULIP [350] on our ensembled shape dataset, but we use the original texts in the four constituent datasets along with the official codebase without backbone scaling. We train OpenShape and ULIP on three different sets of training shapes: "**Ensembled**" denotes using all shapes from the four datasets; "**Ensembled (no LVIS)**" is the same but excludes all shapes from the Objavserse-LVIS subset;

"**ShapeNet**" only includes shapes from the ShapeNet [22] dataset. Note that even when LVIS shapes are included in the training shapes (i.e., the "Ensembled" dataset), their test-time category labels are probably not included in their training texts.

Table 5.2 shows the results. We observe that OpenShape consistently outperforms prior approaches, even when trained only on ShapeNet. When models are trained on our larger-scale ensembled dataset, they receive a significant performance boost. In this case, OpenShape still surpasses retrained ULIP by a significant margin, demonstrating the advantages of our text enrichment, backbone scaling, and other training strategies. Specifically, OpenShape greatly improves the classification accuracy on the long tail categories in Objaverse-LVIS from a dull $< 10\%$ to $\mathbf{46.8}\%$, outperforming the retrained ULIP by about 20 points and reaching a decent top-5 accuracy of $\mathbf{77.0}\%$. These results demonstrate OpenShape's capability to recognize open-world objects effectively. As for ModelNet40, OpenShape achieves a $\mathbf{85.3}\%$ accuracy, surpassing previous methods by a substantial margin of at least 20 percent. OpenShape also achieves impressive top-3 and top-5 accuracies of $\mathbf{96.5}\%$ and $\mathbf{98.0}\%$. To the best of our knowledge, this is the first time zero-shot methods have matched the performance of a fully-supervised 3D learning method on ModelNet40, where OpenShape outperforms fully-supervised 3D ShapeNets [333] and VoxNet [194]. In addition, on ScanObjectNN, which contains challenging real scans with noise and occlusion, OpenShape exhibits decent sim-to-real transfer capabilities. To contextualize, OpenShape-SparseConv achieves $\mathbf{56.7}\%$ zero-shot accuracy on ScanObjectNN without specific sim-to-real training, which surpasses 52.7% reported by SKPConv [321], a recent method specially designed for sim-to-real transfer in point cloud classification tasks.

5.3.2 Few-Shot Linear Probing

In the literature, linear probing is a common way to assess the representation learning capabilities of a model. To perform linear probing, we gather and freeze the representation vectors from all samples in a dataset. Subsequently, we train a linear classifier using these fixed vectors and few-shot class labels. We evaluate the accuracy of the linear classifier on three

Table 5.2. Zero-shot classification on Objaverse-LVIS [50], ModelNet40 [333], and ScanObjectNN [289].

Method	training shape source	Objaverse-LVIS [50]			ModelNet40 [333]			ScanObjectNN [290]		
		Top1	Top3	Top5	Top1	Top3	Top5	Top1	Top3	Top5
PointCLIP [383]	2D inferences, no 3D training	1.9	4.1	5.8	19.3	28.6	34.8	10.5	20.8	30.6
PointCLIP v2 [396]		4.7	9.5	12.9	63.6	77.9	85.0	42.2	63.3	74.5
ReCon [230]	ShapeNet	1.1	2.7	3.7	61.2	73.9	78.1	42.3	62.5	75.6
CG3D [91]		5.0	9.5	11.6	48.7	60.7	66.5	42.5	57.3	60.8
CLIP2Point [106]		2.7	5.8	7.9	49.5	71.3	81.2	25.5	44.6	59.4
ULIP-PointBERT (Official) [350]		6.2	13.6	17.9	60.4	79.0	84.4	51.5	71.1	80.2
OpenShape-SparseConv		11.6	21.8	27.1	72.9	87.2	93.0	52.7	72.7	83.6
OpenShape-PointBERT		10.8	20.2	25.0	70.3	86.9	91.3	51.3	69.4	78.4
ULIP-PointBERT (Retrained)	Ensembled (no LVIS)	21.4	38.1	46.0	71.4	84.4	89.2	46.0	66.1	76.4
OpenShape-SparseConv		37.0	58.4	66.9	82.6	95.0	97.5	54.9	76.8	87.0
OpenShape-PointBERT		39.1	60.8	68.9	**85.3**	96.2	97.4	47.2	72.4	84.7
ULIP-PointBERT (Retrained)	Ensembled	26.8	44.8	52.6	75.1	88.1	93.2	51.6	72.5	82.3
OpenShape-SparseConv		43.4	64.8	72.4	83.4	95.6	97.8	**56.7**	78.9	88.6
OpenShape-PointBERT		**46.8**	**69.1**	**77.0**	84.4	**96.5**	**98.0**	52.2	**79.7**	**88.7**

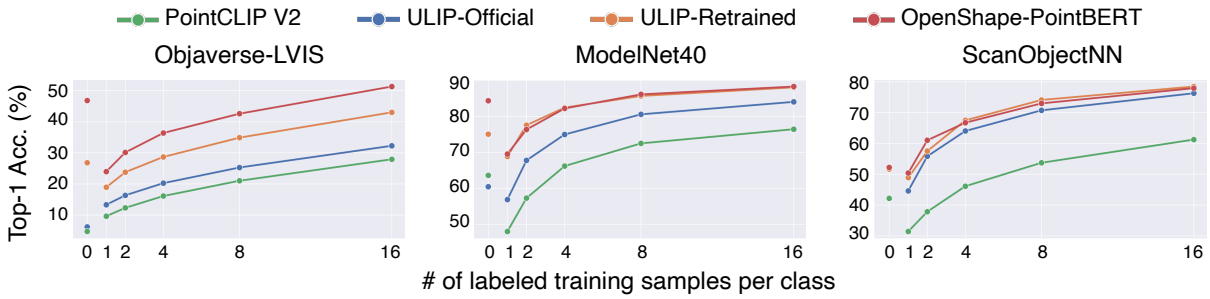


Figure 5.5. Few-shot linear probing on Objaverse-LVIS [50], ModelNet40 [333], and ScanObjectNN [289]. We report the average performance over 10 random seeds.

benchmarks: Objaverse-LVIS [50], ModelNet40 [333], and ScanObjectNN [290]. Figure 5.5 summarizes the performance of OpenShape in comparison with ULIP [350] (official release and our retrained versions) and PointCLIPv2 [396]. On the most challenging Objaverse-LVIS benchmark, OpenShape outperforms all other methods by a large margin. Notably, zero-shot OpenShape beats few-shot linear probes of other methods. On ModelNet40 and ScanObjectNN, we do not see a large performance margin between OpenShape and retrained ULIP. We hypothesize that for few-shot ModelNet40, the error is dominated by in-category sample bias rather than the representation quality; while for ScanObjectNN, the domain gap plays a major role. Since both OpenShape and retrained ULIP are exposed to the same source domain of training objects, their few-shot out-of-domain generalization performances tend to be similar.

Table 5.3. Ablation study. Top 1 zero-shot accuracies on ModelNet40 [333] and Objaverse-LVIS [50] are shown.

Variant	O-LVIS	MNet40
No Objaverse shapes	13.9	75.5
Only Objaverse shapes	41.6	79.2
No backbone scale up	31.7	78.7
No caption & retrieval	37.0	82.9
No text filtering	41.4	82.9
No point rgb, only xyz	39.6	83.6
No text contras. learning	23.3	67.4
No image contras. learning	41.0	81.0
Full	42.0	83.1
Full + hard mining	43.4	83.4

5.3.3 Ablation Study

We perform various ablations by training a scaled version of SparseConv [42] on the ensembled dataset and then evaluate it on the Objaverse-LVIS [50] and ModelNet40 [333] zero-shot classification benchmarks, unless otherwise specified. The results are shown in Table 5.3 and Figures 5.6 and 5.7.

Data and Model Scaling. We investigate the impact of training data by ablating (1) without or with only Objaverse shapes (Tab. 5.3) and (2) with different ratios of our ensembled dataset (Fig. 5.6). We observe that training with 1% of our ensembled dataset (about 8.8k shapes) achieves similar or better zero-shot performance than training without Objaverse shapes (about 77.1k shapes), indicating that the diversity of training data is sometimes more crucial than the scale. In addition, we compare the performances between scaled-up and non-scaled-up backbones. From Tab. 5.3, we demonstrate that model scaling plays an essential role when training on our large-scale ensembled dataset (also Fig. 5.4).

Text Filtering and Enrichment. As shown in Tab. 5.3, both text filtering and text enrichment are beneficial for performance. We also investigate the specific text enrichment strategies to use for the SparseConv and PointBERT backbones. In Fig. 5.7, we observe that both image captioning

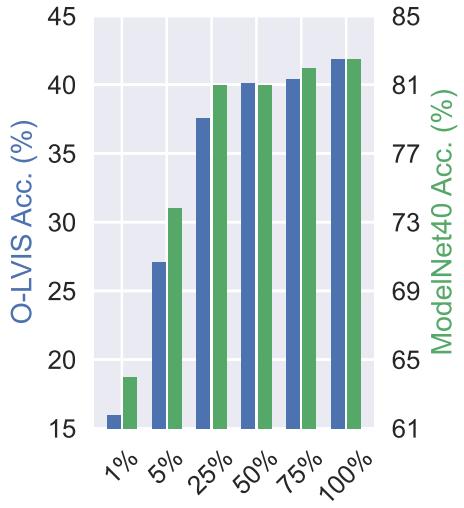


Figure 5.6. Ablation study on using different ratios of training data.

and text retrieval are helpful, and including both yield the best results. Notably, PointBERT improves more than 10 points from text enrichment, highlighting the significance of enhancing text quality.

Other Aspects. We also conduct additional ablation studies on color information, contrastive loss components, and our hard-negative mining strategy in Tab. 5.3. We observe that OpenShape performs well with only *xyz* coordinates as input and no RGB color. While 3D-image contrastive loss is also helpful, we observe that 3D shape-text alignment plays a very essential role for model zero-shot generalization, which necessitates our text filtering and text enrichment strategies that significantly enhance text quality. Lastly, by employing our hard negative mining strategy, OpenShape effectively addresses the issue of unbalanced data distribution, leading to further improvements in performance.

5.3.4 Cross-Modal Applications

Multi-modal 3D Shape Retrieval. Through OpenShape multi-modal representations, we can index and retrieve 3D shapes from images, texts, or point clouds. In this section, we retrieve 3D shapes from our ensembled dataset by calculating the cosine similarity between

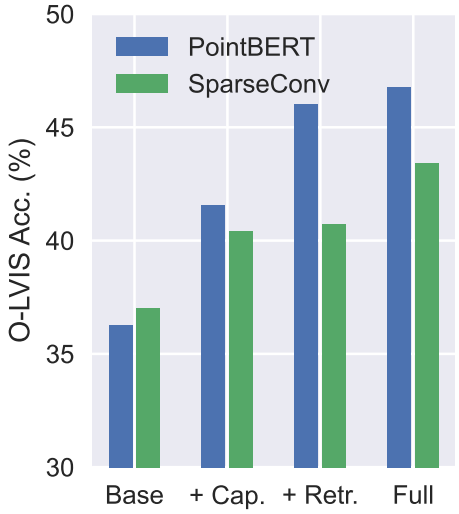


Figure 5.7. Ablation study on different text enrichment strategies.

input embedding(s) and 3D shape embeddings and performing kNN. As shown in Figure 5.8, OpenShape is capable of retrieving visually or semantically similar shapes from a single image or point cloud input. OpenShape embeddings encode a wide range of visual and semantic concepts. In Figure 5.9, we show that OpenShape supports retrieving 3D shapes from detailed text descriptions, which include fine-grained subcategories, attributes, and their combinations. Note that these input texts are typically not present in the raw texts of the retrieved shapes, indicating that OpenShape effectively learns generalizable concepts across shapes. In Figure 5.1, we provide a demo which takes two 3D shapes as inputs and retrieves the shapes that are simultaneously closest to both inputs. This is achieved by finding $\arg\max_i \min(h_i^P \cdot h_a^P, h_i^P \cdot h_b^P)$, where h_a^P and h_b^P denote normalized shape embeddings of the two input shapes. We can see that the retrieved shapes integrate visual or semantic elements in an interesting manner, highlighting the rich concepts and priors encoded in OpenShape embeddings.

Shape-Conditioned Multimodal Generation. As OpenShape's 3D shape representations are aligned with CLIP's image and text embedding spaces, they can serve as inputs into other CLIP-based models to facilitate various multimodal generation applications. For example, we show that by feeding our 3D shape embeddings into ClipCap [208], an off-the-shelf image captioning

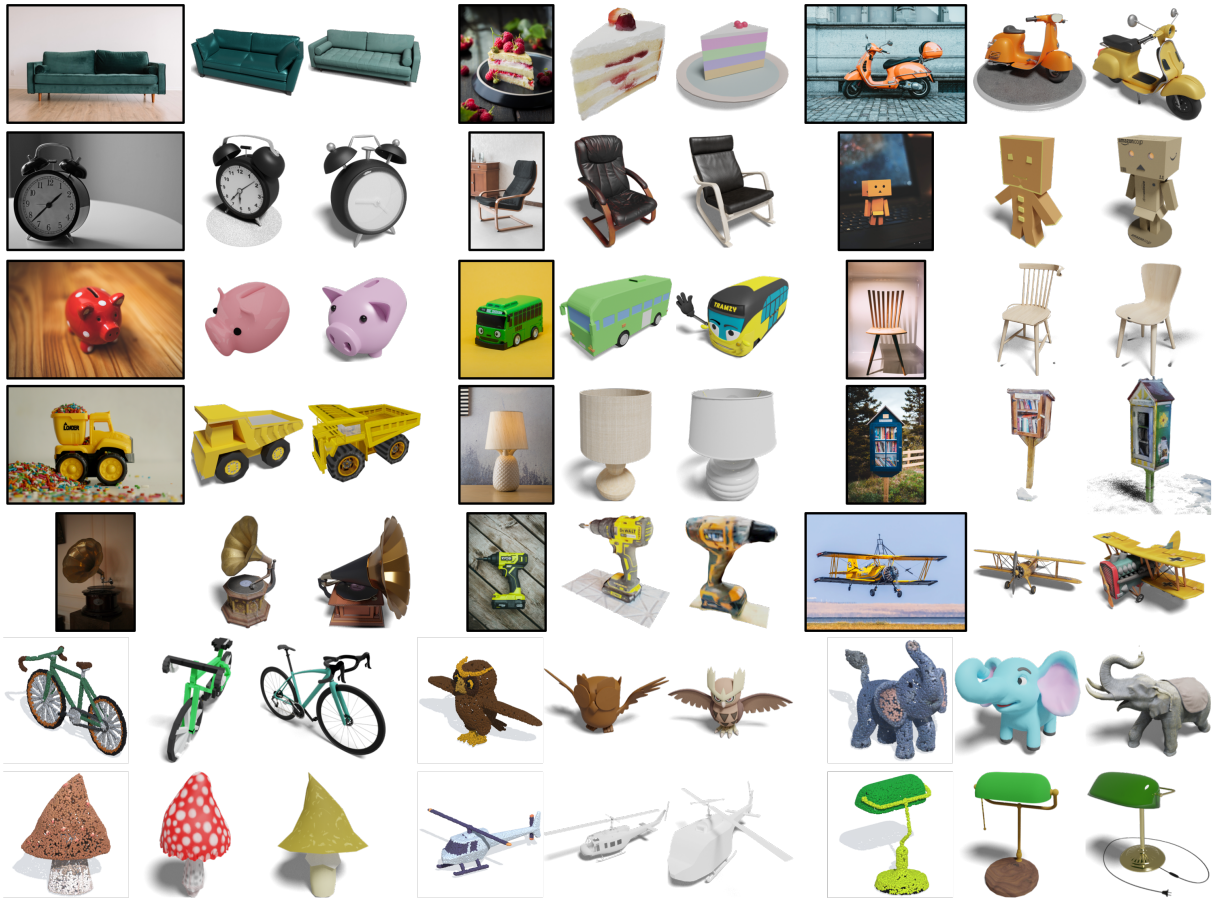


Figure 5.8. 3D shape retrieval from image (first 4 rows) and point cloud (last 2 rows).

model, along with Stable unCLIP [236], a text-to-image diffusion model, we can perform point cloud captioning and point cloud-conditioned image generation (optional text prompt supported) without extra training or finetuning. Qualitative results are shown in Figure 5.10. Please refer to Figure 5.11 for more results.

5.4 Summary

We introduce OpenShape, a novel approach for learning scalable and generalizable multi-modal joint representations for 3D shapes. OpenShape representations effectively capture a wide range of semantic and visual concepts, enabling superior capabilities for open-world 3D shape recognition. By aligning OpenShape with CLIP's embedding space, our shape embeddings

Figure 5.9. Text-input 3D shape retrieval. In each row, we show input texts on the left and two retrieved shapes for each text on the right. OpenShape embedding encodes a wide range of visual and semantic concepts and enables (a) retrieval of fine-grained subcategories (first two rows), and (b) control of attributes (e.g., color, shape, style) and their combinations (last two rows).

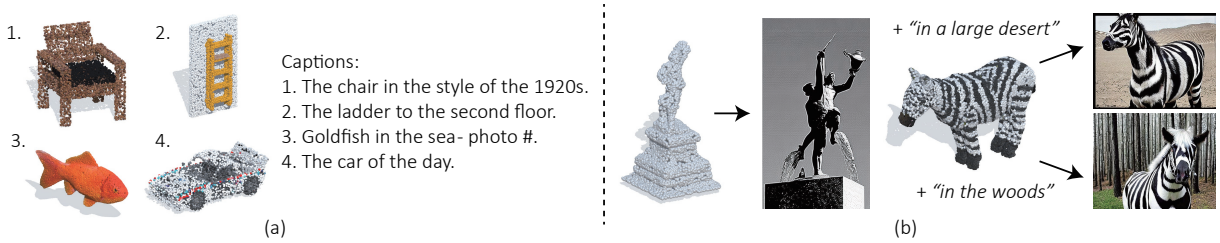


Figure 5.10. (a) Point cloud captioning. (b) Point cloud-conditioned image generation. Our learned 3D shape embeddings can be integrated with off-the-shelf pretrained CLIP-based models (e.g., captioning and image generation models) to support various cross-modal applications.

can be integrated with off-the-shelf CLIP-based models for various cross-modality applications. Moving forward, there are several directions worth further exploration: (a) More 3D data. While we utilized 876k 3D shapes during training, this is still quite limited compared to the 2D counterparts. We hope that our work inspires future investments in more resources to build even more powerful 3D representations. (b) Part-level information. Our current shape representations mainly focus on global semantic and visual features, and it would be beneficial to add more part-level supervision during training. (c) Sim-to-real domain gap. Our model is mainly trained on synthetic data, and it's challenging but crucial to explore explicit designs for reducing the domain gap with real-world shapes.

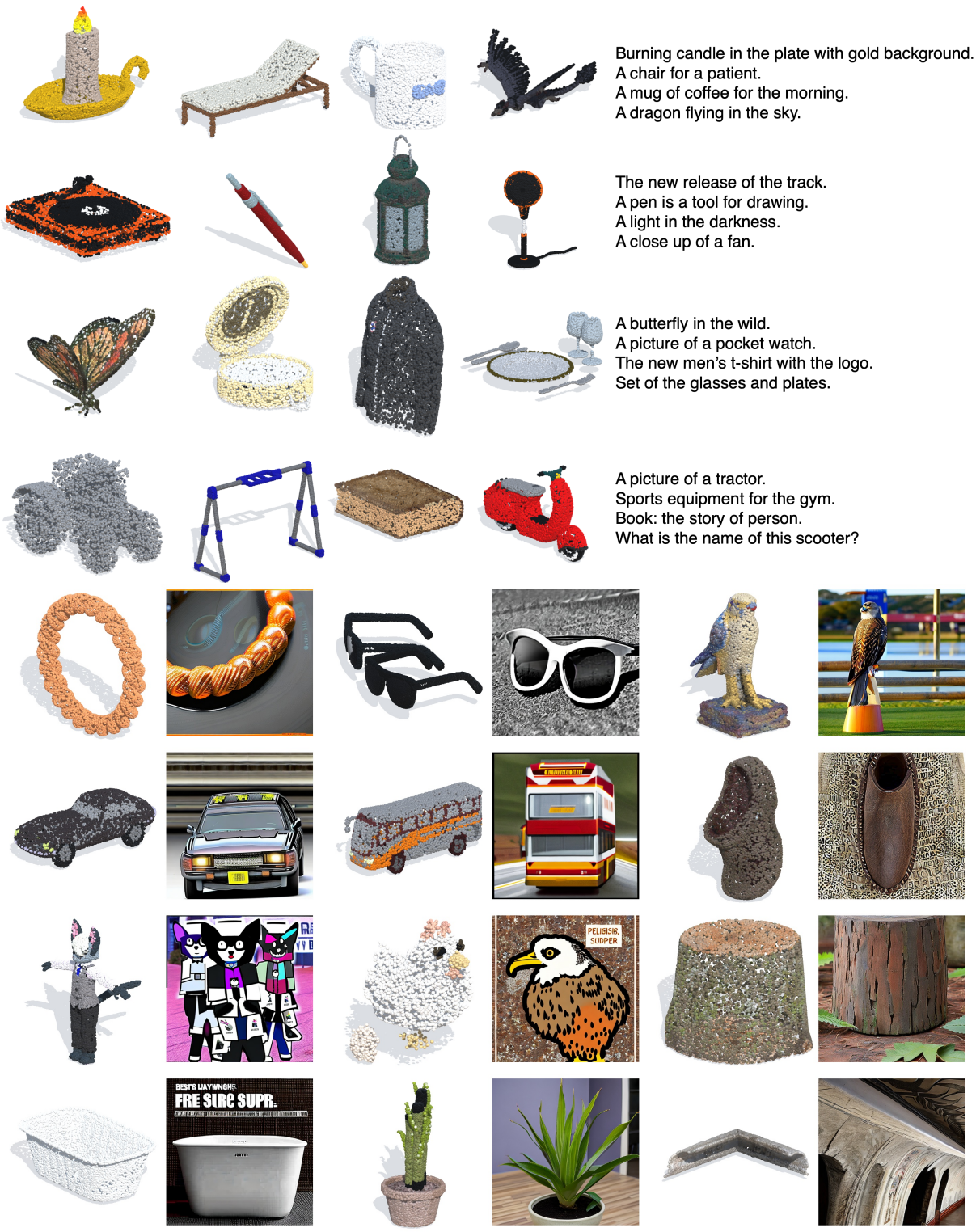


Figure 5.11. More examples of point cloud captioning and point cloud-conditioned image generation.

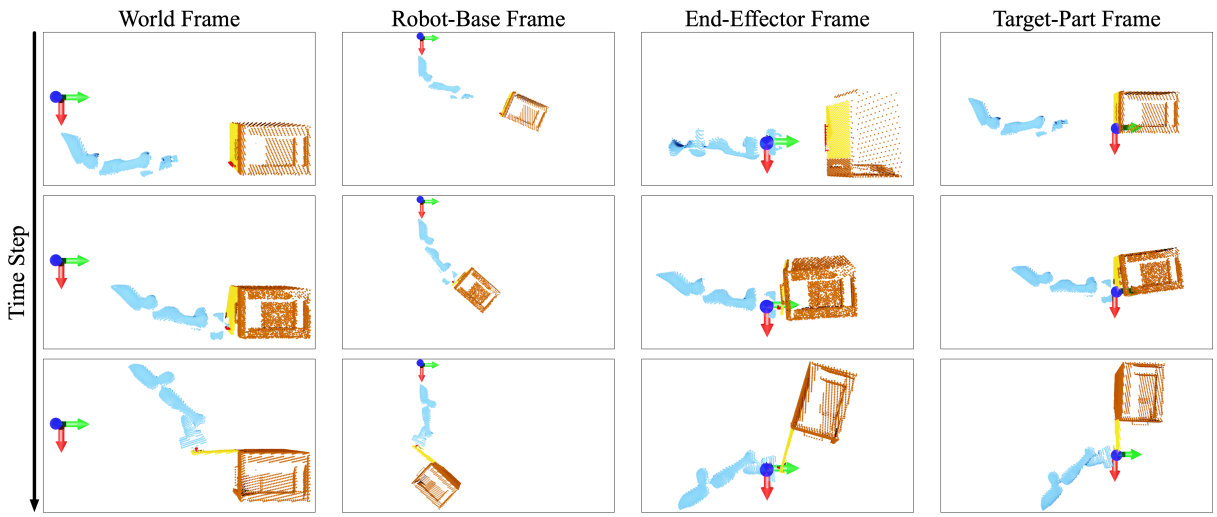


Figure 5.12. Illustration of four coordinate frames, which provide different alignments across time steps. We visualize three point clouds (three time steps) of an OpenCabinetDoor trajectory. Each row shows the same point cloud represented in different coordinate frames. Please zoom in for details. Robot arm, cabinet door handle, cabinet door, and cabinet body are colored in blue, red, yellow, and brown, respectively. RGB arrows indicate the corresponding origin and axes for each frame. Since the point clouds used for policy learning can be rather sparse, we show dense point clouds here for better visualization.

5.5 Related Project: Coordinate Frame Learning for 3D Point Clouds

OpenShape learns a powerful multi-modal representation for 3D point clouds, which is particularly useful when working with 3D point clouds that describe a complete, single static object. However, when dealing with raw point clouds that capture dynamic scenes with multiple interacting objects, the question of how to encode the entire scene and efficiently extract relationships between the objects becomes intriguing. We find that the choice of input point cloud coordinate frame may play a critical role.

Specifically, 3D point cloud started to be used as the input to deep reinforcement learning (RL) for object manipulation [31, 107, 332], which aims at learning mappings directly from raw 3D sensor observations of unstructured environments to robot action commands. When building an agent with point cloud input, existing works [31, 107, 332] typically incorporate off-the-shelf

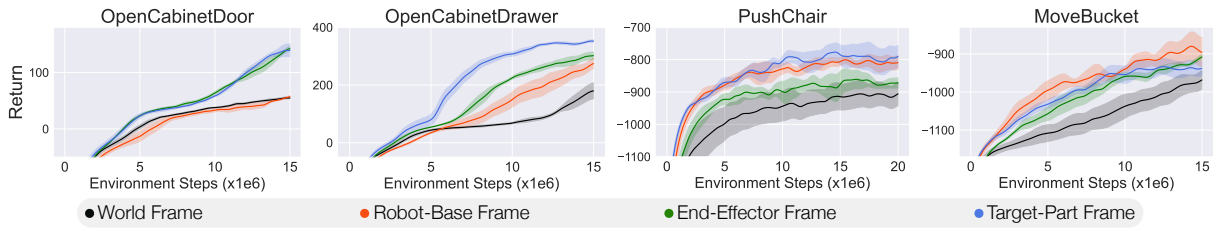


Figure 5.13. Comparison of four coordinate frames on five fully-physical manipulation tasks. The (fused) point cloud is transformed to a *single* coordinate frame before being fed to the visual backbone network. For dual-arm tasks (i.e., PushChair and MoveBucket), we use the right-hand frame as the end-effector frame. For PickObject, which has a fixed base, the world frame is the same as the robot-base frame. Mean and standard deviation over 5 seeds are shown.

point cloud backbone networks (e.g., PointNet [228]) into the pipeline as a feature extractor of the scene. However, some facets in constructing point cloud representations have been overlooked. For example, in the literature of 3D deep learning, the choice of coordinate frame significantly affects task performance [8, 52, 72, 155, 228, 394]. On 3D instance segmentation benchmarks for autonomous driving, previous work such as [227] showed a pipeline to process input point clouds in the camera frame, frustum frame, and object frame subsequently, leading to a large performance boost in comparison to using the camera frame alone. For our goal of manipulation skill learning, point clouds describe dynamic interactions between robots and objects, including frequent contacts and occlusions. This is a novel and more complex setting that differs from well-explored scenarios in 3D supervised learning (e.g., single objects, outdoor scenes for autonomous driving). Under this setting, choices of coordinate frames are more flexible and diverse as multiple entities (e.g., robot and manipulated object) and dynamic movements are involved.

In this project, we first examine whether and how different coordinate frames may impact the performance and sample efficiency of point cloud-based RL for object manipulation tasks. We study four candidate coordinate frames: world frame, robot-base frame, end-effector frame, and target-part frame. These frames differ in positions of origin and orientations of axes, and canonicalize inputs in different manners (e.g., a fixed third-view, ego-centric, hand-centric, object-centric). Please refer to Figure 5.12 for an illustration. The comparison and analysis

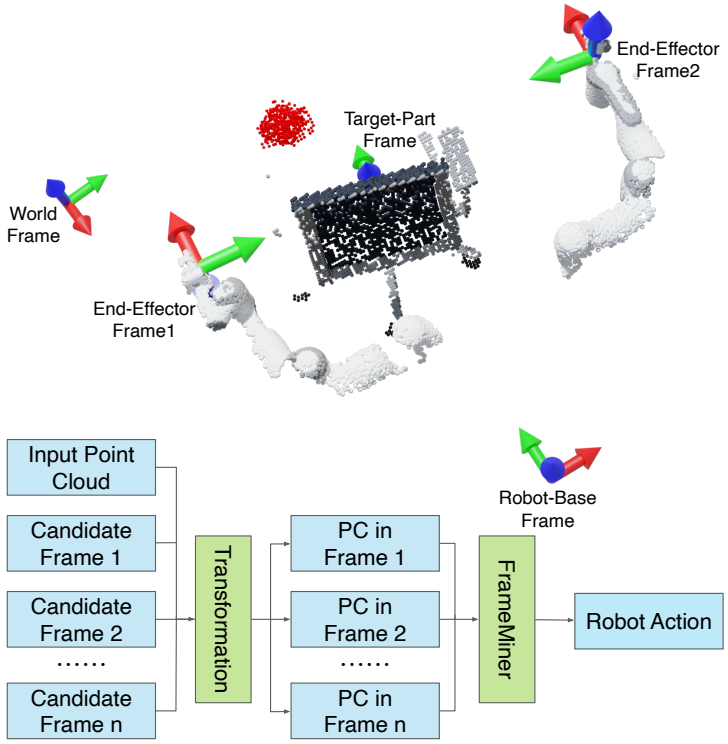


Figure 5.14. A 3D point cloud of a dual-arm robot pushing a chair, which can be represented in various coordinate frames without changing camera placements or requiring extra camera views. Our FrameMiner takes as input a point cloud represented in multiple candidate frames and adaptively fuses their merits, resulting in better performance.

are performed on five distinct physical manipulation tasks adapted from ManiSkill [209] and OCRTOC [182], covering various numbers of arms, robot mobilities, and camera settings. As shown in Figure 5.13, the choice of frames has profound effects. In particular, the end-effector frame and the target-part frames, rarely considered in previous works, lead to significantly better sample efficiency and final convergence than the widely used world frame and robot-base frame on many tasks. Visualization and analysis indicate that, by using different coordinate frames to represent input point clouds, we are actually performing various alignments of input scenes through $\mathbb{SE}(3)$ transformations, which may simplify the learning of visual modules.

However, the well-performing single coordinate frame may vary from task to task, and in many cases, we may need coordination between decisions made according to multiple coordinate frames. For example, tasks equipped with dual-arm robots may benefit from both

left-hand and right-hand frames. For mobile manipulation tasks involving both navigation and manipulation, different frames could favor different skills (e.g., robot-base frame for navigation skills, end-effector frame for manipulation skills). We thus propose three task-agnostic strategies to adaptively select from multiple candidate coordinate frames and fuse their merits, leading to more efficient and effective object manipulation policy learning. Please refer to Figure 5.14 for an overview. Because we do not need to capture additional camera views or rely on task-specific frame selections, our frame mining strategies can be used as a free lunch to improve existing methods on point cloud-based policy learning. We call these fusion approaches as *FrameMiners*. Experimentally, we find that it matters to fuse information from multiple frames, but the specific FrameMiner to choose does not create much performance difference. In particular, we use one of the FrameMiners, MixAction, to interpret the importance of different frames in the policy execution process, and the interpretation agrees with our intuitions.

Chapter 5 incorporates material from the publication "OpenShape: Scaling Up 3D Shape Representation Towards Open-World Understanding", by Minghua Liu, Ruoxi Shi, Kaiming Kuang, Yinhao Zhu, Xuanlin Li, Shizhong Han, Hong Cai, Fatih Porikli, and Hao Su, published in *Advances in Neural Information Processing Systems (NeurIPS 2023)*. The dissertation author was primary investigator and the lead author of this paper.

Chapter 6

Open-World 3D Understanding: Low-Shot Part Segmentation

In the last chapter, we introduced OpenShape. While it is adept at learning powerful cross-modal joint representations for 3D point clouds, it learns a global embedding for each 3D shape, which may fail to accurately capture fine-grained part-level structures. However, human visual perception can parse objects into parts and generalize to unseen objects, which is crucial for understanding their structure, semantics, mobility, and functionality. 3D part segmentation plays a critical role in enabling machines with this ability, facilitating a wide range of applications such as robotic manipulation, AR/VR, and shape analysis and synthesis [6, 158, 205, 346].

Recent part-annotated 3D shape datasets [206, 334, 364] have promoted advances in designing various data-driven approaches for 3D part segmentation [174, 231, 312, 365]. While standard supervised training enables these methods to achieve remarkable results, they often struggle with out-of-distribution test shapes (e.g., unseen classes). However, compared to image datasets, these 3D part-annotated datasets are still orders of magnitude smaller in scale, since building 3D models and annotating fine-grained 3D object parts are laborious and time-consuming. It is thus challenging to provide sufficient training data covering all object categories. For example, the recent PartNet dataset [206] contains only 24 object categories, far less than what an intelligent agent would encounter in the real world.

To design a generalizable 3D part segmentation module, many recent works have focused

Figure 6.1. We propose PartSLIP, a zero/few-shot method for 3D point cloud part segmentation by leveraging pretrained image-language models. The figure shows text prompts and corresponding semantic segmentation results (zoom in for details). Our method also supports part-level instance segmentation. See Figure 6.6 and Figure 6.9 for more results.

on the few-shot setting, assuming only a few 3D shapes of each category during training. They design various strategies to learn better representations, and complement vanilla supervised learning [172, 261, 270, 299, 389]. While they show improvements over the original pipeline, there is still a large gap between what these models can do and what downstream applications need. The problem of generalizable 3D part segmentation is still far from being solved. Another parallel line of work focuses on learning the concept of universal object parts and decomposing a 3D shape into a set of (hierarchical) fine-grained parts [190, 309, 369]. However, these works do not consider the semantic labeling of parts and may be limited in practical use.

In this chapter, we seek to solve the low-shot (zero- and few-shot) 3D part segmentation problem by leveraging pretrained image-language models, inspired by their recent striking performances in low-shot learning. By pretraining on large-scale image-text pairs, image-language models [5, 115, 144, 234, 236, 249, 377] learn a wide range of visual concepts and knowledge, which can be referenced by natural language. Thanks to their impressive zero-shot capabilities, they have already enabled a variety of 2D/3D vision and language tasks [47, 96, 111, 239, 248, 252, 383].

As shown in Figure 6.1, our method takes a 3D point cloud and a text prompt as input,

and generates both 3D semantic and instance segmentations in a zero-shot or few-shot fashion. Specifically, we integrate the GLIP [144] model, which is pretrained on 2D visual grounding and detection tasks with over 27M image-text pairs and has a strong capability to recognize object parts. To connect our 3D input with the 2D GLIP model, we render multi-view 2D images for the point cloud, which are then fed into the GLIP model together with a text prompt containing part names of interest. The GLIP model then detects parts of interest for each 2D view and outputs detection results in the form of 2D bounding boxes. Since it is non-trivial to convert 2D boxes back to 3D, we propose a novel 3D voting and grouping module to fuse the multi-view 2D bounding boxes and generate 3D instance segmentation for the input point cloud. Also, the pretrained GLIP model may not fully understand our definition of parts only through text prompts. We find that an effective solution is prompt tuning with few-shot segmented 3D shapes. In prompt tuning, we learn an offset feature vector for the language embedding of each part name while fixing the parameters of the pretrained GLIP model. Moreover, we propose a multi-view visual feature aggregation module to fuse the information of multiple 2D views, so that the GLIP model can have a better global understanding of the input 3D shape instead of predicting bounding boxes from each isolated 2D view.

To better understand the generalizability of various approaches and their performances in low-shot settings, we propose a benchmark PartNet-Ensembled (PartNetE) by incorporating two existing datasets PartNet [206] and PartNetMobility [334]. Through extensive evaluation on PartNetE, we show that our method enables excellent zero-shot 3D part segmentation. With few-shot prompt tuning, our method not only outperforms existing few-shot approaches by a large margin but also achieves highly competitive performance compared to the fully supervised counterpart. We also demonstrate that our method can be directly applied to iPhone-scanned point clouds without significant domain gaps. In summary, our contributions mainly include:

- We introduce a novel 3D part segmentation method that leverages pretrained image-language models and achieves outstanding zero-shot and few-shot performance.

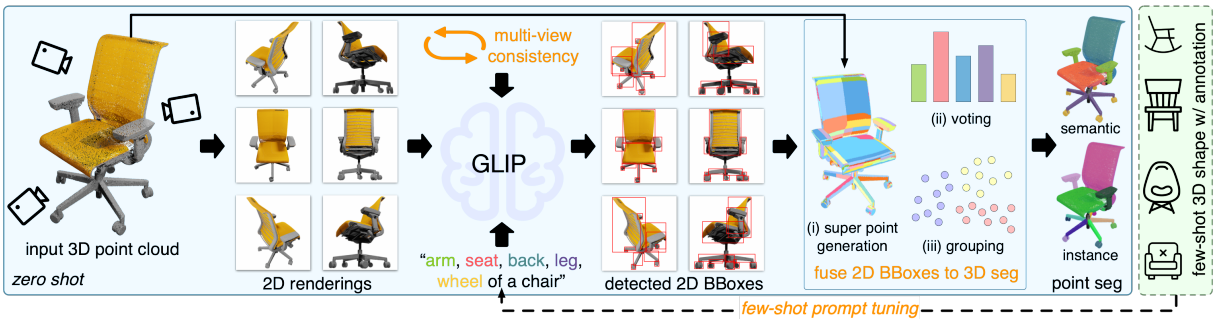


Figure 6.2. The figure shows our overall pipeline. Our proposed components are highlighted in orange.

- We present a 3D voting and grouping module, which effectively converts multi-view 2D bounding boxes into 3D semantic and instance segmentation.
- We utilize few-shot prompt tuning and multi-view feature aggregation to boost GLIP's detection performance.
- We propose a benchmark PartNetE that benefits future work on low-shot and text-driven 3D part segmentation.

6.1 Related Work

6.1.1 3D Part Segmentation

3D part segmentation involves two main tasks: semantic segmentation and instance segmentation. Most 3D backbone networks [229, 231, 283, 312] are capable of semantic segmentation by predicting a semantic label for each geometric primitive (e.g., point or voxel). Existing learning-based approaches solve instance segmentation by incorporating various grouping [44, 89, 118, 157, 293, 308, 310, 376] or region proposal [99, 353, 365] strategies into the pipeline. Different from standard training with per-point part labels, some works leverage weak supervision, such as bounding box [39, 173], language reference game [134], or IKEA manual [306]. Instead of focusing on single objects, [17, 216] also consider part segmentation for scene-scale input. Moreover, unlike the two classical tasks of semantic and instance segmentation,

another parallel line of works decomposes a 3D shape into a set of (hierarchical) fine-grained parts but without considering semantic labels [190, 309, 369], which differs from our objective. Recently, some works also propose to learn a continuous implicit semantic field [132, 393].

6.1.2 Data-Efficient 3D Segmentation

In order to train a generalizable 3D part segmentation network with low-shot data, many existing efforts focus on leveraging various pretext tasks and auxiliary losses [7, 65, 88, 259, 281]. In addition, [86, 212] studies the compositional generalization of 3D parts. [299] deforms input shapes to align with few-shot template shapes. [261] leverages 2D contrastive learning by projecting 3D shapes and learning dense multi-view correspondences. [34] leverages branched autoencoders to co-segment a collection of shapes. Also, some works aim to learn better representations by utilizing prototype learning [389], reinforcement learning [172], and data augmentation [270]. Moreover, there is a line of work investigating label-efficient 3D segmentation [100, 167, 181, 347, 354, 386, 386, 387], assuming a small portion of training data is annotated (e.g., 0.1% point labels). While the setting may be useful in indoor and autonomous driving scenarios, it is not aligned with our goal since the number of training shapes is already limited in our setup.

6.1.3 3D Learning with Image-Language Models

Pretrained image-language models have recently made great strides by pretraining on large-scale image-text pairs [5, 115, 144, 234, 236, 249, 377]. Due to their learned rich visual concepts and impressive zero-shot capabilities, they have been applied to a wide range of 3D vision tasks, such as 3D avatar generation and manipulation [20, 96, 114], general 3D shape generation [110, 129, 201, 252], low-shot 3D shape classification [383], neural radiance fields [111, 294], 3D visual grounding [47, 284], and 3D representation learning [248]. To the best of our knowledge, we are one of the first to utilize pretrained image-language models to help with the task of 3D part segmentation.

6.2 Proposed Method: PartSLIP

6.2.1 Overview: 3D Part Segmentation with GLIP

We aim to solve both semantic and instance segmentation for 3D object parts by leveraging pretrained image-language models (ILMs). There are various large-scale ILMs emerged in the past few years. In order to enable generalizable 3D object part segmentation, the pre-trained ILM is expected to be capable of generating region-level output (e.g., 2D segmentation or 2D bounding boxes) and recognizing object parts. After comparing several released pretrained ILMs (e.g., CLIP [234]), we find that the GLIP [144] model is a good choice. The GLIP [144] model focuses on 2D visual grounding and detection tasks. It takes as input a free-form text description and a 2D image, and locates all phrases of the text by outputting multiple 2D bounding boxes for the input image. By pretraining on large-scale image-text pairs (e.g., 27M grounding data), the GLIP model learns a wide range of visual concepts (e.g., object parts) and enables open-vocabulary 2D detection.

Figure 6.2 shows our overall pipeline, where we take a 3D point cloud as input. Here, we consider point clouds from unprojecting and fusing multiple RGB-D images, which is a common setup in real-world applications and leads to dense points with color and normal. To connect the 2D GLIP model with our 3D point cloud input, we render the point cloud from K predefined camera poses. The camera poses are uniformly spaced around the input point cloud, aiming to cover all regions of the shape. Since we assume a dense and colored point cloud input[1], we render the point cloud by simple rasterization without introducing significant artifacts. The K rendered images are then fed separately into the pretrained GLIP model along with a text prompt. We format the text prompt by concatenating all part names of interest and the object category. For example, for a chair point cloud, the text prompt could be "arm, back, seat, leg, wheel of a chair". Please note that unlike the traditional segmentation networks, which are limited to

[1]Recent commodity-grade 3D scanning devices (e.g., iPhone 12 Pro) can already capture high-quality point clouds (see Figure 6.9).

a closed set of part categories, our method is more flexible and can include any part name in the text prompt. For each 2D rendered image, the GLIP model is expected to predict multiple bounding boxes, based on the text prompt, for all part instances that appear. We then fuse all bounding boxes from K views into 3D to generate semantic and instance segmentation for the input point cloud (Section 6.2.2).

The above pipeline introduces an intuitive zero-shot approach for 3D part segmentation without requiring any 3D training. However, its performance may be limited by the GLIP predictions. We thus propose two additional components, which could be incorporated into the above pipeline to encourage more accurate GLIP prediction: (a) prompt tuning with few-shot 3D data, which enables the GLIP model to quickly adapt to the meaning of each part name (Section 6.2.3); (b) multi-view feature aggregation, which allows the GLIP model to have a more comprehensive visual understanding of the input 3D shape (Section 6.2.4).

6.2.2 Detected 2D BBoxes to 3D Point Segmentation

Although the correspondence between 2D pixels and 3D points are available, there are still two main challenges when converting the detected 2D bounding boxes to 3D point segmentation. First, bounding boxes are not as precise as point-wise labels. A 2D bounding box may cover points from other part instances as well. Also, although each bounding box may indicate a part instance, we are not provided with their relations across views. It's not very straightforward to determine which sets of 2D bounding boxes indicate the same 3D part instance.

Therefore, we propose a learning-free module to convert the GLIP predictions to 3D point segmentation, which mainly includes three steps: (a) oversegment the input point cloud into a collection of super points; (b) assign a semantic label for each super point by 3D voting; and (c) group super points within each part category into instances based on their similarity of bounding box coverage.

3D Super Point Generation: We follow the method in [138] to oversegment the input point

cloud into a collection of super points. Specifically, we utilize point normal and color as features and solve a *generalized minimal partition problem* with an l_0-cut pursuit algorithm [137]. Since points in each generated super point share similar geometry and appearance, we assume they belong to one part instance. The super point partition serves as an important 3D prior when assigning semantic and instance labels. It also speeds up the label assignment, as the number of super points is orders of magnitude smaller than the number of 3D points.

3D Semantic Voting: While a single bounding box may cover irrelevant points from other parts, we want to leverage information from multiple views and the super point partition to counteract the effect of irrelevant points. Specifically, for each pair of super point and part category, we calculate a score $s_{i,j}$ measuring the proportion of the ith super point covered by any bounding box of part category j:

$$s_{i,j} = \frac{\sum_k \sum_{p \in SP_i} [\mathrm{VIS}_k(p)][\exists b \in BB_k^j : \mathrm{INS}_b(p)]}{\sum_k \sum_{p \in SP_i} [\mathrm{VIS}_k(p)]}, \tag{6.1}$$

where SP_i indicates the ith super point, $[\cdot]$ is the Iverson bracket, $\mathrm{VIS}_k(p)$ indicates whether the 3D point p is visible in view k, BB_k^j is a list of predicted bounding boxes of category j in view k, and $\mathrm{INS}_b(p)$ indicates whether the projection of point p in view k is inside the bounding box b.

Note that for each view, we only consider visible points since bounding boxes only contain visible portions of each part instance. Both $\mathrm{VIS}_k(p)$ and $\mathrm{INS}_b(p)$ can be computed based on the information from point cloud rasterization. After that, for each super point i, we assign part category j with the highest score $s_{i,j}$ to be its semantic label.

3D Instance Grouping: In order to group the super points into part instances, we first regard each super point as an individual instance and then consider whether to merge each pair of super points. For a pair of super points SP_u and SP_v, we merge them if: (a) they have the same semantic label, (b) they are adjacent in 3D, and (c) for each bounding box, they are either both included or both excluded.

Specifically, for the second criterion, we find the k nearest neighbors for all points within

each super point. If any point in SP_v is among the k nearest neighbors of a point in SP_u, or vice versa, we consider the super points to be adjacent. For the third criterion, we consider bounding boxes from views where both of them are visible:

$$B = \{b \in BB_k | \, \text{VIS}_k(SP_u) \wedge \text{VIS}_k(SP_v)\}, \tag{6.2}$$

where $\text{VIS}_k(SP_u)$ indicates whether the super point SP_u can be (partially) visible in view k and BB_k indicates all predicted bounding boxes of view k. Suppose B contains n bounding boxes. We then construct two n dimensional vectors I_u and I_v, describing the bounding box coverage of SP_u and SP_v. Specifically, $I_u[i]$ is calculated as:

$$I_u[i] = \frac{\sum_{p \in SP_u}[\text{VIS}_{B[i]}(p)][\text{INS}_{B[i]}(p)]}{\sum_{p \in SP_u}[\text{VIS}_{B[i]}(p)]}, \tag{6.3}$$

where $B[i]$ indicates the ith bounding box of B, $\text{VIS}_{B[i]}(p)$ indicates whether p is visible in the corresponding view of $B[i]$, and $\text{INS}_{B[i]}(p)$ indicates whether the projection of p is inside $B[i]$. If $\frac{|I_u - I_v|_1}{max(|I_u|_1, |I_v|_1)}$ is smaller then a predefined threshold τ, we consider they satisfy the third criterion.

After checking all pairs of super points, the super points are divided into multiple connected components, each of which is then considered to be a part instance. We found that our super point-based module works well in practice.

6.2.3 Prompt Tuning w/ Few-Shot 3D Data

In our method, we utilize natural language to refer to a part. However, natural language can be flexible. An object part can be named in multiple ways (e.g., spout and mouth for kettles; caster and wheel for chairs), and the definition of some parts may be ambiguous (see the dispenser in Figure 6.1). We thus hope to finetune the GLIP model using a few 3D shapes with ground truth part segmentation, so that the GLIP model can quickly adapt to the actual definition of the part names in the text prompt.

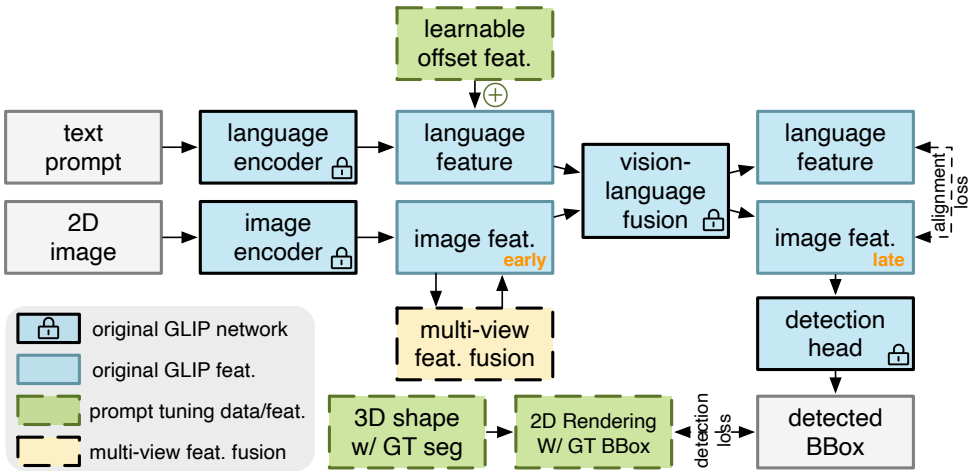


Figure 6.3. The original GLIP pipeline and our proposed additional modules: few-shot prompt tuning and multi-view feature aggregation. We find that early fusion leads to better performance than late fusion.

Figure 6.3 shows the overall architecture of the GLIP model. It first employs a language encoder and an image encoder to extract language features and multi-scale visual features, respectively, which are then fed into a vision-language fusion module to fuse information across modalities. The detection head then takes as input the language-aware image features and predicts 2D bounding boxes. During pretraining, the GLIP network is supervised by both detection loss and image-language alignment loss.

It is not desirable to change the parameters of the visual module or the entire GLIP model since our goal is to leverage only a few 3D shapes for finetuning. Instead, we follow the prompt tuning strategy introduced in GLIP [144] to finetune only the language embedding of each part name while freezing the parameters of the pretrained GLIP model. Specifically, we perform prompt tuning for each object category separately. Suppose the input text of an object category includes l tokens and denote the extracted language features (before VL fusion) as $f_l \in \mathbb{R}^{l \times c}$, where c is the number of channels. We aim to learn offset features $f_o \in \mathbb{R}^{l \times c}$ for f_l and feed their summation $f_l + f_o$ to the remaining GLIP pipeline. The offset features f_o consist of constant vectors for each token (part name), which can be interpreted as a local adjustment of the part definition in the language embedding space. Note that f_o is not predicted by a network but is

directly optimized as a trainable variable during prompt tuning. Also, f_o will be fixed for each object category after prompt tuning.

In order to utilize the detection and alignment losses for optimization, we convert the few-shot 3D shapes with ground truth instance segmentation into 2D images with bounding boxes. Specifically, for each 3D point cloud, we render K 2D images from the predefined camera poses. For generating corresponding 2D ground-truth bounding boxes, we project each part instance from 3D to 2D. Note that, after projection, we need to remove occluded points (i.e., invisible points of each view) and noisy points (i.e., visible but isolated in tiny regions) to generate reasonable bounding boxes. We find that by prompt tuning with only one or a few 3D shapes, the GLIP model can quickly adapt to our part definitions and generalize to other instances.

6.2.4 Multi-View Visual Feature Aggregation

The GLIP model is sensitive to camera views. For example, images taken from some unfamiliar views (e.g., the rear view of a cabinet) can be uninformative and confusing, making it difficult for the GLIP model to predict accurately. However, unlike regular 2D recognition tasks, our input is a 3D point cloud, and there are pixel-wise correspondences between different 2D views. Therefore, we hope the GLIP model can leverage these 3D priors to make better predictions instead of focusing on each view in isolation.

In order to take full advantage of the pretrained GLIP model, we propose a training-free multi-view visual feature aggregation module that could be plugged into the original GLIP network without changing any existing network weights. Specifically, the feature aggregation module takes K feature maps $\{f_k \in \mathbb{R}^{m\times m\times c}\}$ as input, where m is the spatial resolution of the feature map and c is the number of channels. The input feature maps $\{f_k\}$ are generated by the GLIP module separately for each 2D view of the input point cloud. Our feature aggregation module fuses them and generates K fused feature maps $\{f'_k\}$ of the same shape, which are then used to replace the original feature maps and fed into the remaining layers of the GLIP model.

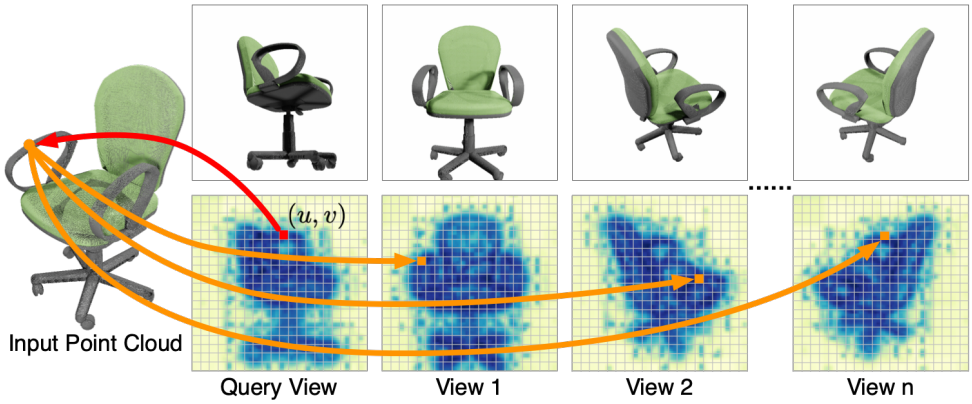


Figure 6.4. Multi-view 2D renderings (first row) and their feature maps (second row). For a feature cell (red), we aggregate all its corresponding feature cells (orange) across views.

As shown in Figure 6.4, for each cell (u,v) of feature map f_i, we find its corresponding cell $(u^{i\to k}, v^{i\to k})$ in each feature map f_k and use their weighted average to serve as the fused feature of the cell:

$$f_i'[u,v] = \frac{1}{\sum_k w_{u,v}^{i\to k}} \sum_k w_{u,v}^{i\to k} f_k[u^{i\to k}, v^{i\to k}]. \tag{6.4}$$

Specifically, we define $P_i(u,v)$ as the set of 3D points that are visible in view i and whose projections lie within cell (u,v). We then choose the cell in view k with the most overlapping 3D points as the corresponding cell: $(u^{i\to k}, v^{i\to k}) = \arg\max_{(x,y)} |P_i(u,v) \cap P_k(x,y)|$ and define the weights $w_{u,v}^{i\to k}$ as $\frac{|P_i(u,v)\cap P_k(u^{i\to k}, v^{i\to k})|}{|P_i(u,v)|}$. Note that if all 3D points in $P_i(u,v)$ are not visible in a view k, then feature map f_k will not contribute to $f_i'[u,v]$. Since the GLIP model generates multi-scale visual features, our aggregation module fuses features of each scale level separately.

There are various options for which visual features to fuse (see Figure 6.3). One intuitive choice is to fuse the final visual features before the detection head, and we denote this choice as *late fusion*. We find that the late fusion does not improve or even degrade the original performance. This is mainly because the final visual features contain too much shape information of the predicted 2D bounding boxes. Directly averaging the final visual features can somehow be seen as averaging bounding boxes in 2D, which does not make sense. Instead, we choose to fuse the visual features before the vision-language fusion (denoted as *early fusion*). Since the text

Table 6.1. The table shows the statistics of the PartNetE dataset: category name, part names, number of few-shot shapes, test shapes, and additional training shapes (if applicable). The 17 overlapping object categories are bolded.

category	parts	few-shot	test	extra-train	category	parts	few-shot	test	extra-train
Bottle	lid	8	49	471	**Microwave**	display, door, handle, button	8	8	234
Box	lid	8	20	0	Mouse	button, cord, wheel	8	6	0
Bucket	handle	8	28	0	Oven	door, knob	8	22	0
Camera	button, lens	8	29	0	Pen	cap, button	8	40	0
Cart	wheel	8	53	0	Phone	lid, button	8	10	0
Chair	arm, back, leg, seat, wheel	8	73	8000	Pliers	leg	8	17	0
Clock	hand	8	23	593	Printer	button	8	21	0
CoffeeMachine	button, container, knob, lid	8	46	0	**Refrigerator**	door, handle	8	36	195
Dishwasher	door, handle	8	40	179	Remote	button	8	41	0
Dispenser	head, lid	8	49	0	Safe	door, switch, button	8	22	0
Display	base, screen, support	8	29	954	**Scissors**	blade, handle, screw	8	39	60
Door	frame, door, handle	8	28	237	Stapler	body, lid	8	15	0
Eyeglasses	body, leg	8	57	0	**StorageFurniture**	door, drawer, handle	8	338	2260
Faucet	spout, switch	8	76	681	Suitcase	handle, wheel	8	16	0
FoldingChair	seat	8	18	0	Switch	switch	8	62	0
Globe	sphere	8	53	0	**Table**	door, drawer, leg, tabletop, wheel, handle	8	93	9799
Kettle	lid, handle, spout	8	21	0	Toaster	button, slider	8	17	0
Keyboard	cord, key	8	29	165	Toilet	lid, seat, button	8	61	0
KitchenPot	lid, handle	8	17	0	**TrashCan**	footpedal, lid, door	8	62	358
Knife	blade	8	36	505	USB	cap, rotation	8	43	0
Lamp	base, body, bulb, shade	8	37	3246	WashingMachine	door, button	8	9	0
Laptop	keyboard, screen, shaft, touchpad, camera	8	47	430	Window	window	8	50	0
Lighter	lid, wheel, button	8	20	0	**45 in total**	**103 in total**	**360**	**1,906**	**28,367**

prompt is not involved yet, the visual features mainly describe the geometry and appearance of the input shape. Fusing these features across views with the 3D priors can thus lead to a more comprehensive visual understanding of the input shape.

6.3 Experiments

6.3.1 Datasets and Metrics

To evaluate the generalizability of various approaches and their performances in the low-shot setting, we curate an ensembled dataset named PartNet-Ensembled (PartNetE), which consists of shapes from existing datasets PartNet [206] and PartNet-Mobility [334]. Note that PartNet-Mobility contains more object categories but fewer shape instances, and PartNet contains more shape instances but fewer object categories. We thus utilize shapes from PartNet-Mobility for few-shot learning and test, and use shapes from PartNet to serve as additional large-scale training data for transfer learning. As a result, the test set of PartNetE contains 1,906 shapes covering 45 object categories. In addition, we randomly reserve 8 shapes from each of the 45 object categories for few-shot training. Also, we may utilize the additional 28,367 shapes from PartNet for training, which cover 17 out of 45 object categories and have

consistent part annotations as the test set. Some of the original part categories in PartNet (e.g., "back_frame_vertical_bar" for chairs) are too fine-grained and ambiguous to evaluate unsupervised text-driven part segmentation approaches. We thus select a subset of 103 parts when constructing the PartNetE dataset, which covers both common coarse-grained parts (e.g., chair back and tabletop) and fine-grained parts (e.g., wheel, handle, button, knob, switch, touchpad) that may be useful in downstream tasks such as robotic manipulation. Please refer to Table 6.1 for the dataset statistics.

We follow [206] to utilize category mIoU and mAP (50% IoU threshold) as the semantic and instance segmentation metrics, respectively. We first calculate mIoU/mAP50 for each part category across all test shapes, and then average part mIoUs/mAP50s that belong to each object category to compute the object category mIoU/mAP50.

6.3.2 Implementation Details

For each 3D shape (i.e., ShapeNet [22] mesh), we use BlenderProc [54] to render 6 views of RGB-D images and segmentation masks with a resolution of 512×512. We unproject the images to the world space to obtain a fused point cloud with colors, normals, and ground truth part labels. The fused point clouds are used as the input for both our method and baseline approaches.

For our method, we render each input point cloud into $K = 10$ color images with Pytorch3D [240]. In few-shot experiments, we utilize 8 point clouds (8×10 rendered images with 2D bounding boxes) of each object category for prompt tuning. The threshold τ in part instance grouping is empirically set to 0.3.

6.3.3 Comparison with Existing Methods

Low-Shot Settings and Baseline Methods

We consider three low-shot settings: (a) zero-shot: no 3D training/finetuning involved; (b) few-shot (45×8): utilize only 8 shapes for each object category during training; (c) few-shot

Table 6.2. Semantic segmentation results on the PartNetE dataset. Object category mIoU(%) are shown. For 17 overlapping object categories, baseline models leverage additional 28k training shapes in the 45x8+28k setting. For the other 28 non-overlapping object categories, there are only 8 shapes per object category during training. Please refer to the supplementary for the full table of all 45 categories.

		Overlapping Categories									Non-Overlapping Categories									
#3D data	method	Bottle	Chair	Display	Door	Knife	Lamp	Storage Furniture	Table	Overall (17)	Camera	Cart	Dis- Penser	Kettle	Kitchen- Pot	Oven	Suit- case	Toaster	Overall (28)	Overll (45)
few-shot w/ extra data (45x8+28k)	PointNet++ [229]	48.8	84.7	78.4	45.7	35.4	68.0	46.9	**63.7**	55.6	6.5	6.4	12.1	20.9	15.8	34.3	40.6	14.7	25.4	36.8
	PointNext [231]	68.4	**91.8**	**89.4**	43.8	58.7	64.9	**68.5**	52.1	**58.5**	33.2	36.3	26.0	45.1	57.0	37.8	13.5	8.3	**45.1**	**50.2**
	SoftGroup [293]	41.4	88.3	62.1	**53.1**	31.3	**82.2**	60.2	54.8	50.2	23.6	23.9	18.9	57.4	45.5	13.6	18.3	26.4	30.7	38.1
few-shot (45x8)	PointNet++ [229]	27.0	42.2	30.2	20.5	22.2	10.5	8.4	7.3	18.1	9.7	11.6	7.0	28.6	31.7	19.4	3.3	0.0	21.8	20.4
	PointNext [231]	67.6	65.1	53.7	46.3	59.7	55.4	20.6	22.1	39.2	26.0	47.7	22.6	60.5	66.0	36.8	14.5	0.0	41.5	40.6
	SoftGroup [293]	20.8	80.5	39.7	16.3	38.3	38.3	18.9	24.9	32.8	28.6	40.8	42.9	60.7	54.8	35.6	29.8	14.8	41.1	38.0
	ACD [65]	22.4	39.0	29.2	18.9	39.6	13.7	7.6	13.5	19.2	10.1	31.5	19.4	40.2	51.8	8.9	13.2	0.0	25.6	23.2
	Prototype [389]	60.1	70.8	67.3	33.4	50.4	38.2	30.2	25.7	41.1	32.0	36.8	53.4	62.7	63.3	36.5	35.5	10.1	46.3	44.3
	Ours	**83.4**	85.3	84.8	40.8	**65.2**	66.0	53.6	42.4	**56.3**	**58.3**	**88.1**	**73.7**	**77.0**	**69.6**	**73.5**	**70.4**	**60.0**	**61.3**	**59.4**
zero-shot	**Ours**	76.3	60.7	43.8	2.7	46.8	37.1	29.4	47.7	31.8	21.4	87.7	16.5	20.8	4.7	33.0	40.2	13.8	24.4	27.2

Table 6.3. Instance segmentation results on the PartNetE dataset. Category mAP50 (%) are shown. See supplementary for the full table.

		Overlapping Categories									Non-Overlapping Categories									
#3D data	method	Bottle	Chair	Display	Door	Knife	Lamp	Storage Furniture	Table	Overall (17)	Camera	Cart	Dis- Penser	Kettle	Kitchen- Pot	Oven	Suit- case	Toaster	Overall (28)	Overll (45)
45x8+28k	PointGroup [118]	38.2	87.6	65.1	**23.4**	19.3	62.7	**49.1**	**46.4**	41.7	8.6	29.2	24.0	61.3	59.4	13.8	15.6	7.0	24.6	31.0
	SoftGroup [293]	43.9	**89.1**	68.7	21.2	27.2	63.3	**49.1**	46.2	**42.4**	0.7	28.4	26.4	63.8	59.3	16.4	13.5	7.5	**25.6**	**31.9**
few-shot (45x8)	PointGroup [118]	8.0	77.2	16.7	3.7	15.6	9.8	0.0	0.0	14.6	4.7	28.5	30.7	52.1	57.0	0.0	0.0	0.0	16.8	16.0
	SoftGroup [293]	22.4	87.7	27.5	5.6	10.3	19.4	11.6	14.2	21.3	11.2	29.8	37.8	63.4	65.7	10.4	8.0	10.7	28.4	25.7
	Ours	**79.4**	84.4	**82.9**	17.9	**43.9**	**68.3**	32.8	32.3	**42.5**	**36.8**	**83.3**	**63.5**	**75.4**	**70.5**	**64.5**	**44.9**	**38.4**	**46.2**	**44.8**
zero-shot	**Ours**	75.5	54.5	32.9	1.3	22.1	35.8	10.9	36.6	20.9	8.4	79.3	9.3	18.3	1.1	25.9	34.2	4.5	16.2	18.0

with additional data ($45 \times 8 + 28k$): utilize 28,367 shapes from PartNet [206] in addition to the 45×8 shapes during training. The 28k shapes cover 17 of the 45 object categories. Here, the last setting ($45 \times 8 + 28k$) describes a realistic setup, where we have large-scale part annotations for some common categories (17 categories in our case) but only a few shapes for the other categories. We aim to examine whether the 28k data of the 17 categories can help the part segmentation of the other 28 underrepresented categories. All settings are tested on the same test set.

We compare with PointNet++ [229] and PointNext [231] for semantic segmentation, and compare with PointGroup [118] and SoftGroup [293] for instance segmentation. We train four baseline approaches on the PartNetE dataset by taking point clouds with normals as input. For semantic segmentation, we follow [206] to sample 10,000 points per shape as network input. For

instance segmentation, we sample up to 50,000 points per shape. For each pair of baseline and setting, we train a single network.

In addition to the four baselines mentioned above, we compare against two methods dedicated to few-shot 3D semantic segmentation: ACD [65] and Prototype [389]. In ACD, we decompose the mesh of each 3D shape into approximate convex components with CoACD [318] and utilize the decomposition results for adding an auxiliary loss to the pipeline of PointNet++. In Prototype, we utilize the learned point features (by PointNext backbone) of few-shot shapes to construct 100 prototypes for each part category, which are then used to classify each point of test shapes.

Evaluation Results

Table 6.2 shows the results of semantic segmentation. Our method achieves impressive zero-shot performance on some common object categories (such as bottle, chair, and table), but also poor performances on certain categories (e.g., kettle). This is mainly due to the pretrained GLIP model may not understand the meaning of the text prompt (e.g., spout for kettles). After prompt tuning with 8-shot 3D data, our method achieves a 59.4% mIoU and outperforms all baseline methods from the few-shot setting and even the $45 \times 8 + 28k$ setting. For the $45 \times 8 + 28k$ setting, baseline methods are trained with additional 28k shapes covering 17 categories. **For these overlapping categories, it's a fully-supervised setting, but our 8-shot version can achieve highly competitive overall mIoU (56.3% vs. 58.5%).** Note that the 28k training data is of limited help for the baselines to generalize to non-overlapping categories. Our method outperforms all baselines on non-overlapping categories by a large margin. The two few-shot strategies ACD and Prototype improve the performance of the original backbone, but there are still large gaps compared to our method. Please see Figure 6.1 for example results of our methods and see Figure 6.5 for qualitative comparison.

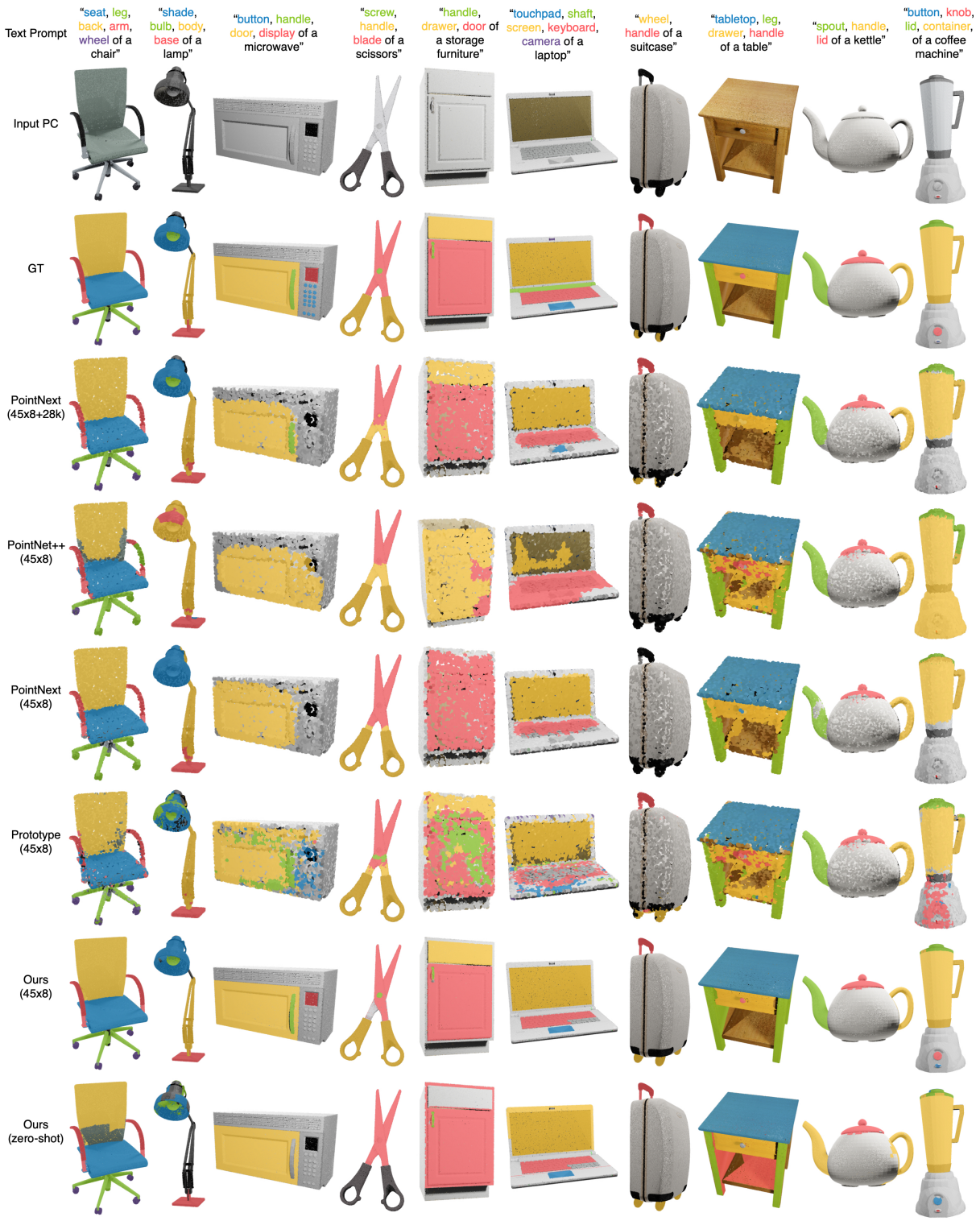


Figure 6.5. Qualitative comparison between our method and baseline approaches on the PartNetE dataset. Semantic segmentation results are shown. For baseline approaches, we randomly sample 10,000 points as input. "45x8" indicates the few-shot setting, where the model is trained with 8 shapes per object category. "45x8+28k" indicates the setting where the additional 28k shapes are used for training .

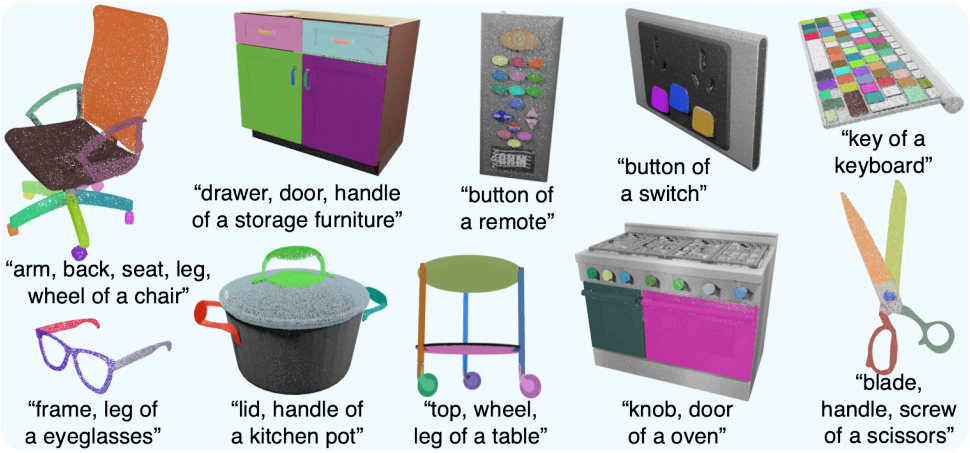


Figure 6.6. Instance segmentation results of our method (8-shot) on the PartNetE dataset. Different part instances are in different colors (zoom in for details).

Table 6.4. Ablation study of the proposed components. We show the performances of both GLIP 2D detection (category mAP50) and 3D semantic segmentation (category mIoU) on three categories. *3D semantic segmentation is generated by assigning part labels to all visible points in bounding boxes.

BBox2 3DSeg	Prompt Tuning	Feat Aggre.	Chair 2D	Chair 3D	Kettle 2D	Kettle 3D	Suitcase 2D	Suitcase 3D	All 3D
			50.4	50.6*	26.4	7.5*	31.9	21.1*	
✓			50.4	60.7	26.4	20.8	31.9	40.2	27.2
✓	✓		80.7	83.8	82.1	72.7	65.6	65.1	58.0
✓		✓	52.3	64.5	32.2	25.9	36.4	49.1	27.7
✓	✓	✓	82.4	85.3	84.3	77.0	68.9	70.4	59.4

Table 6.3 shows the results of instance segmentation. We observe similar phenomena as semantic segmentation. Our method achieves 18.0% mAP50 for the zero-shot setting and 44.8% mAP50 for the 8-shot setting, which outperforms all baseline approaches from both 45×8 and $45 \times 8 + 28k$ settings. See Figure 6.6 for qualitative examples.

6.3.4 Ablation Studies

Proposed Components:

We ablate the proposed components, and the results are shown in Table 6.4. For the first row, we only utilize the pretrained GLIP model. In order to get 3D semantic segmentation,

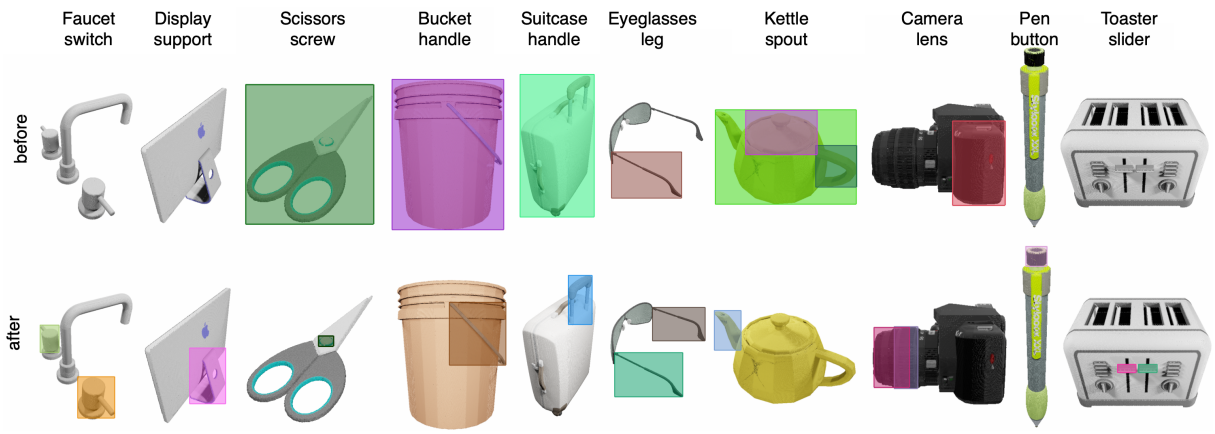


Figure 6.7. Ablation study of few-shot prompt tuning. First row: 2D part detection results of the GLIP pretrained model (zero-shot). Second row: detection results after 8-shot prompt tuning.

we assign part labels to all visible points within bounding boxes. The numbers indicate that this strategy is less effective than our proposed 3D voting and grouping module (second row). Moreover, without our proposed module, we are not able to get 3D instance segmentation. The second and third rows compare the impact of (8-shot) prompt tuning. We observe significant improvements, especially on the Kettle category, as the zero-shot GLIP model fails to understand the meaning of "spout" but it adapts to the definition after few-shot prompt tuning. Please refer to Figure 6.7 for qualitative examples. The second and fourth rows compare our multi-view feature aggregation module. Without utilizing any extra data for finetuning, we leverage multi-view 3D priors to help the GLIP model better understand the input 3D shape and thus improve performance. After integrating all three modules, we achieve the final good performance (last row).

Variations of Input Point Clouds:

Table 6.5 evaluates the robustness of our method about variations of input point clouds. We observe that when the input point cloud is partial and does not cover all regions of the object, our method still performs well (second row). Also, we find that after removing the textures of the ShapeNet models and generating the input point cloud by using gray-scale images, our method can achieve good performance as well, suggesting that textures are less important in recognizing

Table 6.5. Ablation study of various input point clouds. We show the semantic segmentation results of the Chair category.

setting	# views	image reso.	texture	Chair mIoU (%)
original	6	512×512	w/	85.3
partial pc	2	512×512	w/	84.3
no texture	6	512×512	w/o	84.0
sparse pc	6	128×128	w/	82.4
sparse pc	6	64×64	w/	68.3

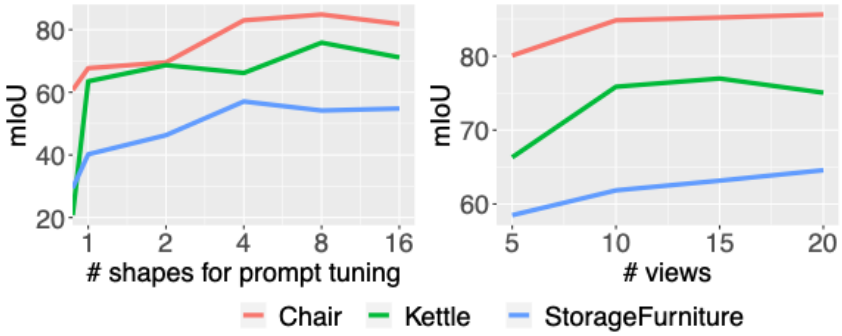


Figure 6.8. Ablation study of the number of shapes in prompt tuning and the number of 2D views (K). Category mIoU of 3D semantic segmentation on the PartNetE dataset are shown.

object parts. However, we find that the performance of our method may degrade when the input point cloud becomes sparse. On the one hand, sparse point clouds cause a larger domain gap for 2D renderings of point clouds. On the other hand, the sparsity makes it hard for our super point generation algorithm to produce good results. That being said, we want to point out that dense point clouds are already mostly available in our daily life (see Section 6.3.5).

Number of Shapes in Prompt Tuning:

We ablate the number of shapes used for prompt tuning, and the results are shown in Figure 6.8 (left). We observe that only using one single shape for prompt tuning can already improve the performance of the pretrained GLIP model a lot in some categories (e.g., Kettle). Also, after using more than 4 shapes, the gain from increasing the number of shapes slows down. We also find that prompt tuning is less effective for object categories that have richer appearance and structure variations (e.g., StorageFurniture).

Table 6.6. Early vs. late fusion in multi-view feature aggregation. We compare GLIP detection (mAP50) on the Suitcase category.

w/o fusion	early fusion	late fusion
65.6	68.9	47.3

Number of 2D Views:

We render $K = 10$ 2D views for each input point cloud in our main experiments. We ablate the value of K, and the results are shown in Figure 6.8 (right). We observe a significant performance drop when K is reduced to 5 and also a mild gain when using a larger K.

Early Fusion vs. Late Fusion:

In the last paragraph of Section 6.2.4, we discuss two choices for multi-view feature aggregation: early fusion and late fusion. Table 6.6 compares these two choices and verifies that late fusion will even degrade the performance while early fusion is helpful.

GLIP vs. CLIP:

We have considered using other pretrained vision-language models, such as CLIP [234]. However, we find that the pretrained CLIP model fails to recognize fine-grained object parts and has difficulty generating region-level output.

6.3.5 Real-World Demo

Thanks to the strong generalizability of the GLIP model, our method can be directly deployed in the real world without a significant domain gap. As shown in Figure 6.9, we use an iPhone 12 Pro Max, equipped with a LiDAR sensor, to capture a video and feed the fused point cloud to our method. We observe similar performances as in our synthetic experiments. Please note that existing 3D networks are sensitive to the input format. For example, they assume objects are normalized in per-category canonical poses. Also, they need to overcome the significant domain gap, making it hard to deploy them directly in real scenarios.

Figure 6.9. Each pair shows a captured point cloud by iPhone (left) and the semantic segmentation result of our method (right).

6.4 Summary

The current pipeline utilizes predicted bounding boxes from the GLIP model. We notice that GLIPv2 [377] has 2D segmentation capabilities, but their pretrained model is not released at the time of submission. We admit that it will be more natural to use 2D segmentation results, which are more accurate than bounding boxes, from pretrained models. However, we want to point out that it is still non-trivial to get 3D instance segmentation even from multi-view 2D segmentation, and all components of our proposed method would still be useful (with necessary adaptations). A bigger concern is that our method cannot handle the interior points of objects. It also suffers from long running time due to point cloud rendering and multiple inferences of the GLIP model. Therefore, using our method to distill the knowledge of 2D VL models and

Figure 6.10. CoACD decompose a solid mesh into a set of components and utilize the convex hulls of the components (shown in different colors) to represent the original shape. Compared to prior works, we can better capture the fine-grained structures of the input shape with fewer components. See handles of the oven and the cabinet, slots of the toaster, and the spout of the kettle (zoom in for details). The high-quality decomposition enables delicate object interaction in downstream applications (e.g., a robot opens the drawer by grabbing the handles).

train 3D foundation models is a promising future direction, which may lead to more efficient inferences.

6.5 Other Related Projects on 3D Segmentation and Decomposition

In addition to open-world 3D semantic part segmentation for objects, I also have two projects related to 3D segmentation and decomposition. The first project aims to decompose a 3D shape into a set of approximately convex components, without any semantic correspondence. The convex properties enable many efficient geometry processing algorithms and benefit various applications, such as physical simulation.

The other project focuses on LiDAR point cloud segmentation for autonomous driving, where we seek to leverage 3D geometric priors during training and reduce the reliance on dense point labels.

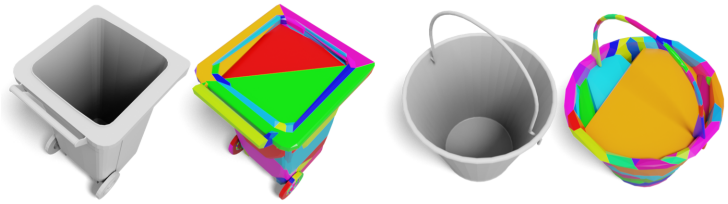


Figure 6.11. Failure cases of the boundary-distance-based methods (from HACD [192]). Focusing only on the boundary distance between the shape and its convex hull, HACD may fail to handle the hollow structures and fill the interior space.

6.5.1 Approximate Convex Decomposition for 3D Meshes

With the development of 3D depth sensors, VR/AR, and physics simulation, large-scale detailed 3D models have become more common. In addition to employing data structures such as octrees and bounding volume hierarchies (BVH) to accelerate specific geometry processing algorithms, another common strategy for handling complex 3D models is to decompose them into simpler components. In particular, convex decomposition has aroused great interest. Many fundamental geometry problems in rendering and physics simulation are non-trivial and computationally expensive to solve for general shapes. However, if input shapes are convex polyhedra, many of them can be formulated as convex optimization problems, and efficient algorithms can be specifically designed. Examples include determining whether a 3D point lies inside or outside of a mesh [268], checking whether two meshes intersect [15, 169, 203], and calculating the minimum distance between two meshes [74].

Decomposing a 3D solid shape into a minimum number of exact convex components is the *exact convex decomposition* (ECD) problem, which has proven to be NP-hard [23, 219]. Although many suboptimal heuristics [24] have been proposed, they usually output a large number of small components, preventing them from practical applications. Instead, the *approximate convex decomposition* (ACD) problem [151] proposes to lift the strict convexity constraint and only requires the decomposed components to be approximately convex. Since ACD approaches [151, 192, 193, 285] typically generate a much smaller number of components, whose convex hulls can then be used to approximate the original shape and speed up downstream applications, ACD

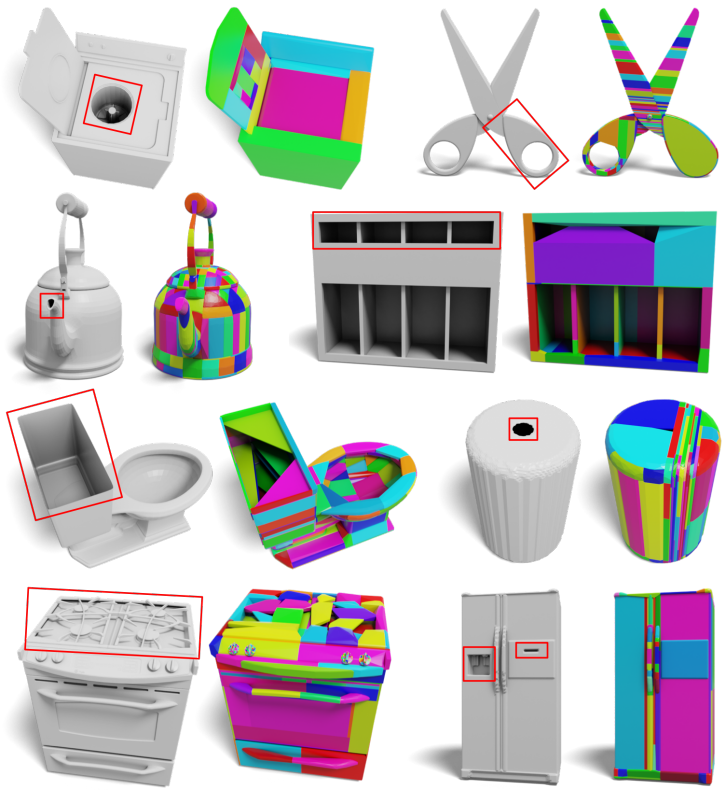


Figure 6.12. Failure cases of the volume-based methods (from V-HACD [193]). Focusing on the volume difference, V-HACD may fill holes when the relative volume of the errors is not too large (e.g., thin planar structures). The red rectangles highlight the error-prone regions.

works have recently received more attention. For example, V-HACD [193] is currently one of the most popular open-source ACD methods and has been adopted by a wide range of game engines and physics simulation SDKs.

Existing ACD methods share a similar overall pipeline. In order to quantify the decomposition quality, they first define a concavity metric to measure the similarity between a decomposed component and its convex hull. They then design a heuristic cost function to decompose the 3D meshes greedily. There are three major shortcomings of existing ACD methods: **(a) Concavity metric:** Prior works mainly utilize two types of metrics: boundary-distance-based concavity [73, 150, 151, 152, 156, 192], which measures the distance between the boundary surfaces of the shape and its convex hull; and volume-based concavity [11, 193, 285], which calculates the volume difference between the solid shape and its convex hull. However, both

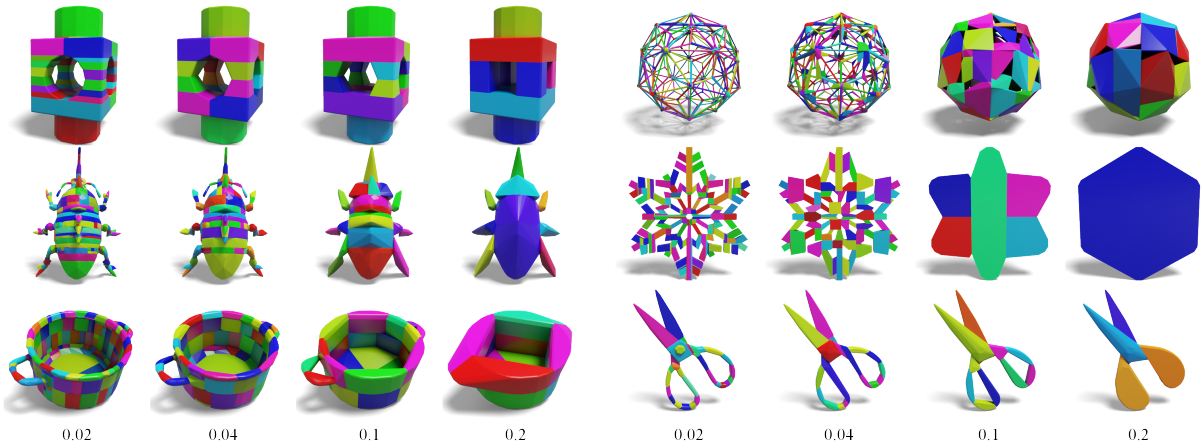


Figure 6.13. Comparison of different concavity thresholds. For each example, we show the decomposition results under different concavity thresholds ranging from 0.02 to 0.2. Users can intuitively balance the level of detail and the number of components by adjusting the concavity threshold ε.

metrics may fail to preserve the collision conditions in some cases, which means some positions in the space are unlikely to collide shape, but collide with the decomposition results. Please refer to Figures 6.11 and 6.12 for some examples. The insensitivity of existing concavity metrics to changes in collision conditions can be fatal for preserving object functionality. For example, they might cause an algorithm to stuff the slots of a toaster. **(b) Component representation:** There are two common strategies for representing components and decomposing shapes. The first one is to decompose shapes by grouping the triangle faces [151, 156, 192], which results in zig-zag boundaries of the components and intersecting convex hulls. In contrast, V-HACD [193] first voxelizes the input mesh and then decomposes the voxels. However, the voxelization introduces discretization artifacts, which even makes the algorithm unable to recognize already convex shapes. **(c) One-step greedy search:** Most previous works [193, 285] decompose the shapes by recursively performing locally optimal actions. They often take short-sighted actions and end up generating more components. Furthermore, considering only one step may lead to various corner cases, which requires different heuristic terms [193] as workarounds. Please see Figure 6.14 for a comparison.

In this project, we introduce a novel approximate convex decomposition method for 3D

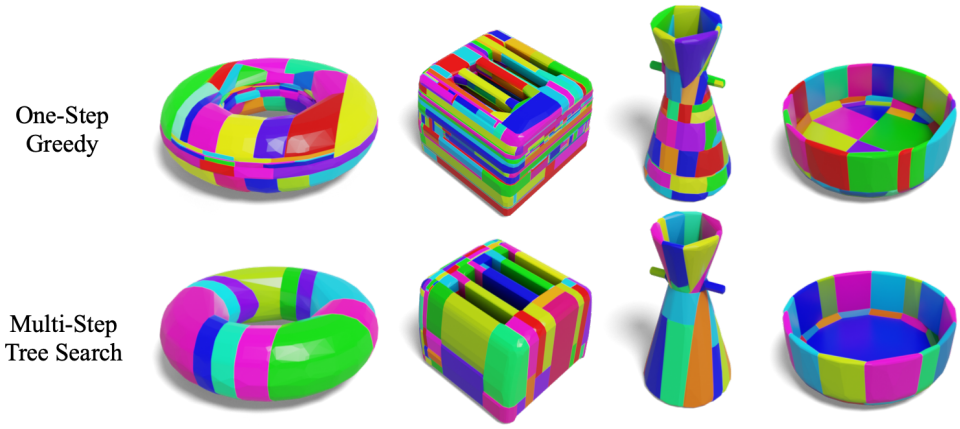


Figure 6.14. First row: results from a one-step greedy strategy with our proposed concavity metric. Second row: results from our method with multi-step tree search. Both methods are tested with the same concavity threshold.

meshes, which effectively addresses the limitations of existing approaches from three corresponding perspectives: (a) We propose a novel collision-aware concavity metric that examines the component from both the boundary surface and shape interior by sampling points and calculating Hausdorff distance. The proposed concavity encourages preserving the collision conditions by penalizing the inclusion of regions that are far away from the original shape. We also propose an efficient way to calculate the concavity to speed up the decomposition. (b) We decompose shapes by directly cutting 3D solid meshes with 3D planes, which results in flat boundaries between components. It ensures intersection-free convex hulls and avoids the defects caused by voxelization. We also provide a lightweight mesh cutting implementation, which is 100x faster than off-the-shelf libraries. (c) We propose utilizing the Monte Carlo tree search to determine cutting planes, which simulates and searches multiple future actions before each cutting. Compared to the one-step greedy search, we are more likely to find cutting planes that lead to a better global solution and avoid unnecessary cuttings. In addition, by considering multiple steps, we no longer need various heuristic terms to prevent corner cases.

We evaluate our method on the V-HACD benchmark [193] and PartNet Mobility [334], a large-scale articulated object dataset. We show that our method better preserves the collision conditions and accurately approximates the fine-grained structures (e.g., drawer handles, kettle

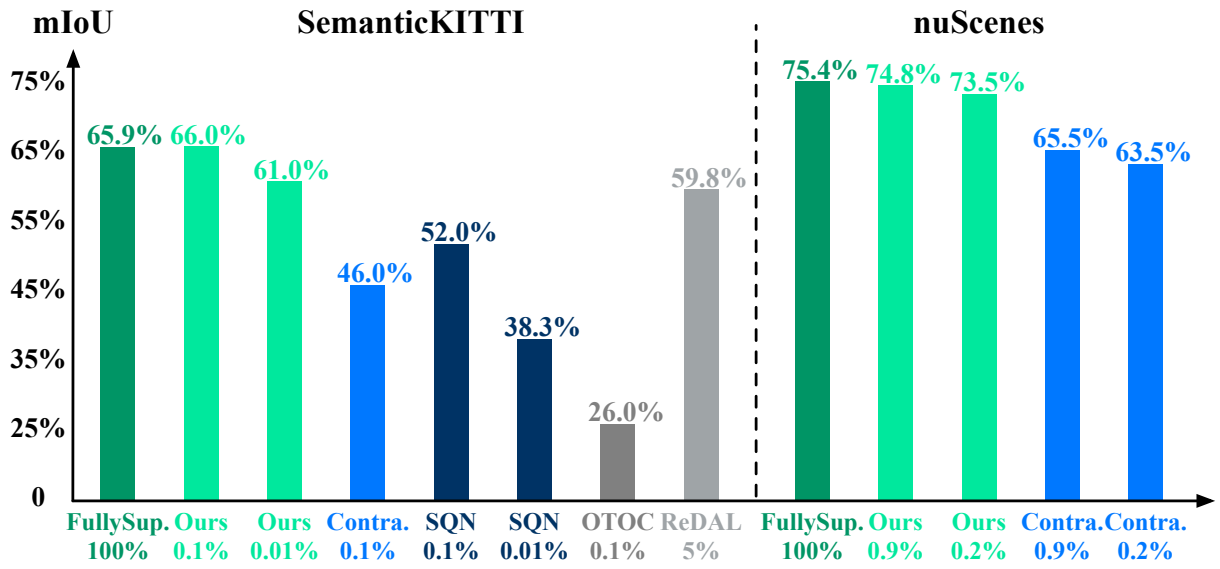


Figure 6.15. We compare LESS with Cylinder3D [397] (our fully-supervised counterpart), ContrastiveSceneContext [100], SQN [103], OneThingOneClick [181], and ReDAL [331] on the SemanticKITTI [14] and nuScenes [19] validation sets. The ratio between labels used and all points is listed below each bar. Please note that all competing label-efficient methods mainly focus on indoor settings and are not specially designed for outdoor LiDAR segmentation.

spouts, inner rings of scissors, and toaster slots) with fewer convex components. Our decomposition results thus enable delicate and fast object interaction in downstream applications. See Figures 6.10 and 6.13 for some examples.

6.5.2 Label-Efficient Semantic Segmentation for LiDAR Point Clouds

Light detection and ranging (LiDAR) sensors have become a necessity for most autonomous vehicles. They capture more precise depth measurements and are more robust against various lighting conditions compared to visual cameras. Semantic segmentation for LiDAR point clouds is an indispensable technology as it provides fine-grained scene understanding, complementary to object detection. For example, semantic segmentation help self-driving cars distinguish drivable and non-drivable road surfaces and reason about their functionalities, like parking areas and sidewalks, which is beyond the scope of modern object detectors.

Based on large-scale public driving-scene datasets [14, 19], several LiDAR semantic segmentation approaches have recently been developed [37, 275, 343, 352, 397]. Typically, these

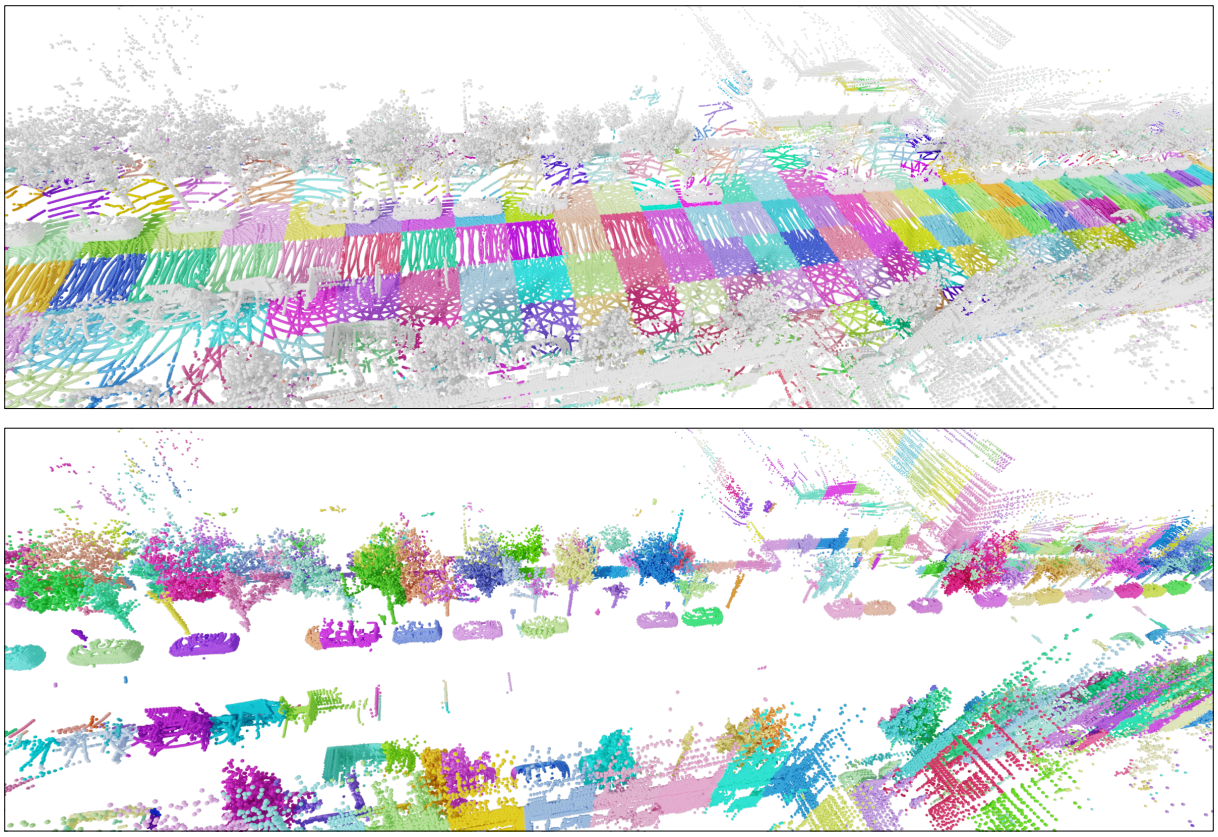


Figure 6.16. Examples of the pre-segmentation results. First row: detected ground points of each cell. Non-ground points are colored in gray. Each other color indicates a proposed ground component. Second row: connected components of the non-ground points. Each color indicates a connected component. The example is from the nuScenes dataset, where 40 scans are fused.

methods require fully labeled point clouds during training. Since a LiDAR sensor may perceive millions of points per second, exhaustively labeling all points is extremely laborious and time-consuming. Moreover, it may fail to scale when we extend the operational domain (e.g., various cities and weather conditions) and seek to cover more rare cases. Therefore, to scale up the system, it is critical to have label-efficient approaches for LiDAR semantic segmentation, whose goal is to minimize the quantity of human annotations while still achieving high performance.

While there are some prior works studying label-efficient semantic segmentation, they mostly focus on indoor scenes [10, 48] or 3D object parts [22], which are quite different in point cloud appearance and object type distribution, compared to the outdoor driving scenes (e.g., significant variances in point density, extremely unbalanced point counts between common types,

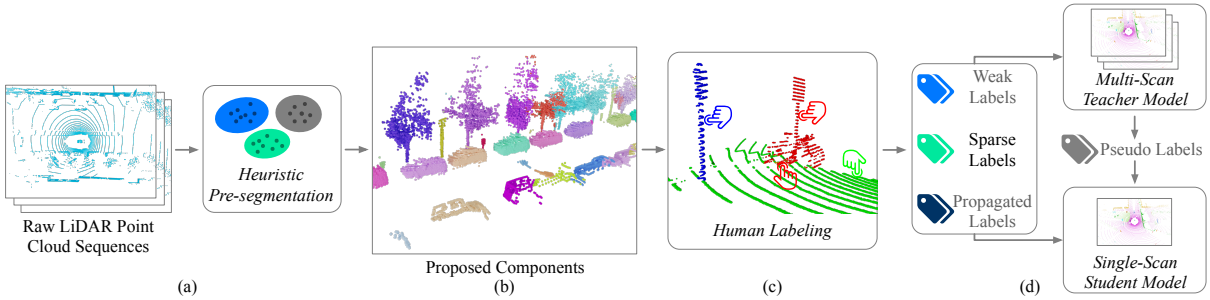


Figure 6.17. Overview of our LESS pipeline. (a) We first utilize a heuristic algorithm to pre-segment each LiDAR sequence into a set of connected components. (b) Examples of the proposed components. Different colors indicate different components. For clear visualization, components of ground points are not shown. (c) Human annotators only need to coarsely label each component. Each color denotes a proposed component, and each click icon indicates a labeled point. Only sparse labels are directly annotated by humans. (d) We then train the network to digest various labels and utilize multi-scan distillation to exploit richer semantics in the temporally fused point clouds.

like ground and vehicles, and less common ones, such as cyclists and pedestrians). Besides, most prior explorations tend to address the problem from two independent perspectives, which may be less effective in our outdoor setting. Specifically, one perspective is improving labeling efficiency, where the methods resort to active learning [189, 263, 331], weak labels [244, 316], and 2D supervision [295] to reduce labeling efforts. The other perspective focuses on training, where the efforts assume the partial labels are given and design semi/weakly supervised learning algorithms to exploit the limited labels and strive for better performance [81, 181, 189, 244, 344, 347, 389].

We proposes a novel framework, label-efficient semantic segmentation (LESS), for LiDAR point clouds captured by self-driving cars. Different from prior works, our method co-designs the labeling process and the model learning, as shown in Figure 6.17. Our co-design is based on two principles: 1) the labeling step is designed to provide bare minimum supervision, which is suitable for state-of-the-art semi/weakly supervised segmentation methods; 2) the model training step can tap into the labeling policy as a prior and deduce more learning targets. The proposed method can fit in a straightforward way with most state-of-the-art LiDAR segmentation backbones without introducing any network architectural change or extra computational complexity when deployed onboard. Our approach is suitable for effectively labeling and learning

from scratch. It is also highly compatible with mining long-tail instances, where, in practice, we mainly want to identify and annotate rare cases based on trained models.

Specifically, we leverage a philosophy that outdoor-scene objects are often well-separated when isolating ground points and design a heuristic approach to pre-segment an outdoor scene into a set of connected components. Please refer to Figure 6.16 for an example. The component proposals are of high purity (i.e., only contain one or a few classes) and cover most of the points. Then, instead of meticulously labeling all points, the annotators are only required to label one point per class for each component. In the model learning process, we train the backbone segmentation network with the sparse labels directly annotated by humans as well as the derived labels based on component proposals. To encourage a more descriptive embedding space, we employ contrastive prototype learning [70, 143, 181, 267, 357], which increases intra-class similarity and inter-class separation. We also leverage a multi-scan teacher model to exploit richer semantics within the temporally fused point clouds and distill the knowledge to boost the performance of the single-scan model.

We evaluate the proposed method on two large-scale autonomous driving datasets, SemanticKITTI [14] and nuScenes [19]. We show that our method significantly outperforms existing label-efficient methods (see Figure 6.15). With extremely limited human annotations, such as 0.1% labeled points, the approach achieves highly competitive performance compared to the fully supervised counterpart, demonstrating the potential of practical deployment.

Chapter 6 incorporates material from the publication "PartSLIP: Low-Shot Part Segmentation for 3D Point Clouds via Pretrained Image-Language Models", by Minghua Liu, Yinhao Zhu, Hong Cai, Shizhong Han, Zhan Ling, Fatih Porikli, and Hao Su, published in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2023)*. The dissertation author was primary investigator and the lead author of this paper.

Chapter 7

Conclusion and Open Problems

7.1 Conclusion

In this dissertation, we have thoroughly investigated the challenges and opportunities in 3D object generation and understanding within an open-world context. Our work has significantly advanced the state of the art by leveraging large-scale pre-trained 2D vision models and the latest expansive 3D datasets, addressing the limitations of traditional methods that were confined to a narrow range of object categories. Through our research, we have enabled more generalizable and scalable approaches, empowering 3D generation and understanding systems to handle a diverse array of open-world 3D objects.

In the domain of open-world 3D generation, we introduced One-2-3-45, a pioneering method that generates 3D objects from a single image through a rapid feed-forward process that takes less than a minute. Our innovative approach integrates 2D diffusion models to predict multi-view images from a single view, followed by a sparse-view reconstruction model that transforms these images into 3D representations. This method addresses both generalizability and speed, setting a new paradigm in the field and inspiring subsequent research. The two major challenges in this direction—namely, the 3D inconsistency of multi-view predictions and the quality of the feed-forward reconstruction module—were further addressed in our follow-up works. In One-2-3-45++, we improved the consistency of predicted multi-view images and incorporated a multi-view conditioned 3D diffusion model to reduce reliance on consistent multi-

view inputs. In MeshFormer, we enhanced the sparse-view reconstruction model by integrating various 3D native priors, leading to significantly improved training efficiency and the generation of 3D geometries with fine-grained, sharp details.

In the area of open-world 3D understanding, our efforts focused on both the global and local attributes of 3D shapes. We explored two strategies to leverage priors from 2D models: training a 3D-native network through knowledge distillation and applying inference-only multi-view fusion. In the OpenShape project, we developed a cross-modal joint representation for 3D point clouds by distilling knowledge from 2D vision models into a 3D-native encoder. This work emphasizes scaling up cross-modal representation alignment, resulting in robust 3D representations capable of encoding a wide range of visual concepts and knowledge. On a more localized level, our PartSLIP project addressed the challenge of low-shot part segmentation for 3D point clouds. By utilizing the pre-trained 2D model GLIP and employing 2D images as intermediaries, we extended the model's capabilities from 2D to 3D. We devised a novel algorithm to merge multi-view predictions and integrated multiple strategies to incorporate 3D priors into the 2D model, enhancing cross-view consistency and accuracy. Extensive evaluations demonstrated that PartSLIP achieves remarkable zero-shot performance, with its few-shot version rivaling or surpassing fully-supervised methods.

Through these contributions, this dissertation has provided new insights and practical approaches for advancing 3D generation and understanding in open-world scenarios. Our work not only extends the applicability of 3D deep learning methods to a broader range of objects but also paves the way for future research in this rapidly evolving field.

7.2 Open Problems

Despite the advancements made in this dissertation, several open problems remain, warranting further exploration. For open-world 3D generation, the journey forward involves addressing several critical challenges:

High-Quality Physically-Based Materials: The generation of high-quality, physically-based materials remains a challenge, particularly due to the scarcity of 3D models with high-quality textures. Current methods often rely on baked textures rather than disentangled materials, which limits their practical application.

Human Editability: Current AI-generated 3D shapes are often holistic, limiting artists' ability to refine or edit these shapes. Future methods should focus on generating part structures to enable human editing. Additionally, ensuring that the topology of the mesh and UV unwrapping are regular and artist-friendly will benefit post-processing.

Controllability: There is a need for more control and diverse input options beyond a single image or text, such as sparse views, sketches, or style transfer, to generate assets that meet user specifications. Interactive generation and conversion-based editing (e.g., selecting a region to modify) are also promising directions.

Simulation and Interaction: Ensuring that 3D assets can support physics-based digital worlds requires further properties, such as part structure (e.g., opening doors, pushing buttons) and mesh quality (e.g., watertightness). These features are essential for enabling interaction between objects and agents in simulations.

Static to Dynamic Shapes: Extending from static to dynamic shapes presents a significant challenge, necessitating methods that can capture and generate time-varying geometries.

Object-Level to Scene-Level Generation: Transitioning from generating individual objects to complete scenes is a critical next step, requiring models that can capture the complex relationships and interactions within a scene.

For open-world 3D understanding, the future holds several key areas that require further investigation to enhance the robustness and applicability:

Real-World Point Clouds: Current approaches often focus on complete, perfect point clouds sampled from mesh surfaces. Extending these methods to handle real-world captured point clouds, which are often incomplete, noisy, and contain background points, is essential.

Zero-Shot Feed-Forward Models for Part-Level Understanding: PartSLIP currently relies

on few-shot tuning and is a per-shape optimization method requiring long processing times. Developing strong feed-forward models with good zero-shot performance could alleviate this bottleneck.

From Object-Level to Scene-Level: As with generation, extending understanding from individual objects to entire scenes is a significant open problem, demanding models that can understand complex spatial relationships and interactions within a scene.

Integration with LLMs and Human Interaction: Integrating 3D understanding with large language models (LLMs) to enable dialogue and conversation with humans represents an exciting frontier. This could open new avenues for interactive 3D applications and more natural human-computer interactions.

While significant progress has been made, the field of open-world 3D generation and understanding remains rich with challenges and opportunities. Addressing these challenges will require not only advancements in computational methods but also the development of more sophisticated datasets, models, and evaluation metrics. As the field evolves, interdisciplinary collaboration will be crucial in pushing the boundaries of what is possible, ultimately leading to more robust, versatile, and human-interpretable 3D systems. The open problems outlined here provide a roadmap for future research, guiding the next generation of innovations in 3D deep learning.

Bibliography

[1] Ahmed Abdelreheem, Ivan Skorokhodov, Maks Ovsjanikov, and Peter Wonka. Satr: Zero-shot semantic segmentation of 3d shapes. *arXiv preprint arXiv:2304.04909*, 2023.

[2] 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.

[3] Idan Achituve, Haggai Maron, and Gal Chechik. Self-supervised learning for domain adaptation on point clouds. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 123–133, 2021.

[4] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3d point clouds. In *International conference on machine learning*, pages 40–49. PMLR, 2018.

[5] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *arXiv preprint arXiv:2204.14198*, 2022.

[6] Jacopo Aleotti and Stefano Caselli. A 3d shape segmentation approach for robot grasping by parts. *Robotics and Autonomous Systems*, 60(3):358–366, 2012.

[7] Antonio Alliegro, Davide Boscaini, and Tatiana Tommasi. Joint supervised and self-supervised learning for 3d real world challenges. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 6718–6725. IEEE, 2021.

[8] Yara Ali Alnaggar, Mohamed Afifi, Karim Amer, and Mohamed ElHelw. Multi projection fusion for real-time semantic segmentation of 3d lidar point clouds. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1800–1809, 2021.

[9] Shivangi Aneja, Justus Thies, Angela Dai, and Matthias Nießner. Clipface: Text-guided editing of textured 3d morphable models. *arXiv preprint arXiv:2212.01406*, 2022.

[10] Iro Armeni, Sasha Sax, Amir R Zamir, and Silvio Savarese. Joint 2d-3d-semantic data for indoor scene understanding. *arXiv preprint arXiv:1702.01105*, 2017.

[11] Marco Attene, Michela Mortara, Michela Spagnuolo, and Bianca Falcidieno. Hierarchical convex approximation of 3d shapes for fast region selection. In *Computer graphics forum*, volume 27, pages 1323–1332. Wiley Online Library, 2008.

[12] Abhishek Badki, Orazio Gallo, Jan Kautz, and Pradeep Sen. Meshlet priors for 3d mesh reconstruction.

arXiv preprint arXiv:2001.01744, 2020.

[13] Romain Beaumont. Clip retrieval: Easily compute clip embeddings and build a clip retrieval system with them. https://github.com/rom1504/clip-retrieval, 2022.

[14] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Jurgen Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 9297–9307, 2019.

[15] Gino van den Bergen. A fast and robust gjk implementation for collision detection of convex objects. *Journal of graphics tools*, 4(2):7–25, 1999.

[16] Fausto Bernardini, Joshua Mittleman, Holly Rushmeier, Cláudio Silva, and Gabriel Taubin. The ball-pivoting algorithm for surface reconstruction. *IEEE transactions on visualization and computer graphics*, 5(4):349–359, 1999.

[17] Alexey Bokhovkin, Vladislav Ishimtsev, Emil Bogomolov, Denis Zorin, Alexey Artemov, Evgeny Burnaev, and Angela Dai. Towards part-based understanding of rgb-d scans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7484–7494, 2021.

[18] G. Bradski. Perspective-n-point (pnp) pose computation (the opencv library). https://docs.opencv.org/4.x/d5/d1f/calib3d_solvePnP.html, 2000.

[19] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11621–11631, 2020.

[20] Zehranaz Canfes, M Furkan Atasoy, Alara Dirik, and Pinar Yanardag. Text and image guided 3d avatar generation and manipulation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 4421–4431, 2023.

[21] Eric R Chan, Koki Nagano, Matthew A Chan, Alexander W Bergman, Jeong Joon Park, Axel Levy, Miika Aittala, Shalini De Mello, Tero Karras, and Gordon Wetzstein. Genvs: Generative novel view synthesis with 3d-aware diffusion models, 2023.

[22] 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.

[23] Bernard Chazelle, David P Dobkin, Nadia Shouraboura, and Ayellet Tal. Strategies for polyhedral surface decomposition: An experimental study. *Computational Geometry*, 7(5-6):327–342, 1997.

[24] Bernard M Chazelle. Convex decompositions of polyhedra. In *Proceedings of the thirteenth annual ACM symposium on Theory of computing*, pages 70–79, 1981.

[25] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *European Conference on Computer Vision*, pages 333–350. Springer, 2022.

[26] Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14124–14133, 2021.

[27] Dave Zhenyu Chen, Yawar Siddiqui, Hsin-Ying Lee, Sergey Tulyakov, and Matthias Nießner. Text2tex: Text-driven texture synthesis via diffusion models. *arXiv preprint arXiv:2303.11396*, 2023.

[28] Guanying Chen, Kai Han, Boxin Shi, Yasuyuki Matsushita, and Kwan-Yee K. Wong. Sdps-net: Self-calibrating deep photometric stereo networks. In *CVPR*, 2019.

[29] Rui Chen, Yongwei Chen, Ningxin Jiao, and Kui Jia. Fantasia3d: Disentangling geometry and appearance for high-quality text-to-3d content creation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2023.

[30] Runnan Chen, Youquan Liu, Lingdong Kong, Xinge Zhu, Yuexin Ma, Yikang Li, Yuenan Hou, Yu Qiao, and Wenping Wang. Clip2scene: Towards label-efficient 3d scene understanding by clip. *arXiv preprint arXiv:2301.04926*, 2023.

[31] Tao Chen, Jie Xu, and Pulkit Agrawal. A system for general in-hand object re-orientation. In *Conference on Robot Learning*, pages 297–307. PMLR, 2022.

[32] Yiwen Chen, Tong He, Di Huang, Weicai Ye, Sijin Chen, Jiaxiang Tang, Xin Chen, Zhongang Cai, Lei Yang, Gang Yu, et al. Meshanything: Artist-created mesh generation with autoregressive transformers. *arXiv preprint arXiv:2406.10163*, 2024.

[33] Zhiqin Chen, Andrea Tagliasacchi, and Hao Zhang. Bsp-net: Generating compact meshes via binary space partitioning. *arXiv preprint arXiv:1911.06971*, 2019.

[34] Zhiqin Chen, Kangxue Yin, Matthew Fisher, Siddhartha Chaudhuri, and Hao Zhang. Bae-net: Branched autoencoder for shape co-segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8490–8499, 2019.

[35] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5939–5948, 2019.

[36] Zilong Chen, Feng Wang, and Huaping Liu. Text-to-3d using gaussian splatting. *arXiv preprint arXiv:2309.16585*, 2023.

[37] Ran Cheng, Ryan Razani, Ehsan Taghavi, Enxu Li, and Bingbing Liu. Af2-s3net: Attentive feature fusion with adaptive feature selection for sparse semantic segmentation network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12547–12556, 2021.

[38] Yen-Chi Cheng, Hsin-Ying Lee, Sergey Tulyakov, Alexander G Schwing, and Liang-Yan Gui. Sdfusion: Multimodal 3d shape completion, reconstruction, and generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4456–4465, 2023.

[39] Julian Chibane, Francis Engelmann, Tuan Anh Tran, and Gerard Pons-Moll. Box2mask: Weakly supervised 3d semantic instance segmentation using bounding boxes. In *European Conference on Computer Vision*, pages 681–699. Springer, 2022.

[40] Han-Pang Chiu, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. Automatic class-specific 3d reconstruction from a single image. *CSAIL*, pages 1–9, 2009.

[41] Gene Chou, Yuval Bahat, and Felix Heide. Diffusion-sdf: Conditional generative modeling of signed distance functions. 2023.

[42] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3075–3084, 2019.

[43] Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VIII 14*, pages 628–644. Springer, 2016.

[44] Ruihang Chu, Yukang Chen, Tao Kong, Lu Qi, and Lei Li. Icm-3d: Instantiated category modeling for 3d instance segmentation. *IEEE Robotics and Automation Letters*, 7(1):57–64, 2021.

[45] Chin Seng Chua and Ray Jarvis. Point signatures: A new representation for 3d object recognition. *International Journal of Computer Vision*, 25:63–85, 1997.

[46] Jasmine Collins, Shubham Goel, Kenan Deng, Achleshwar Luthra, Leon Xu, Erhan Gundogdu, Xi Zhang, Tomas F Yago Vicente, Thomas Dideriksen, Himanshu Arora, et al. Abo: Dataset and benchmarks for real-world 3d object understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21126–21136, 2022.

[47] Rodolfo Corona, Shizhan Zhu, Dan Klein, and Trevor Darrell. Voxel-informed language grounding. *arXiv preprint arXiv:2205.09710*, 2022.

[48] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5828–5839, 2017.

[49] Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, et al. Objaverse-xl: A universe of 10m+ 3d objects. *arXiv preprint arXiv:2307.05663*, 2023.

[50] 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.

[51] Congyue Deng, Chiyu Jiang, Charles R Qi, Xinchen Yan, Yin Zhou, Leonidas Guibas, Dragomir Anguelov, et al. Nerdi: Single-view nerf synthesis with language-guided diffusion as general image priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20637–20647, 2023.

[52] Congyue Deng, Or Litany, Yueqi Duan, Adrien Poulenard, Andrea Tagliasacchi, and Leonidas J Guibas. Vector neurons: A general framework for so (3)-equivariant networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12200–12209, 2021.

[53] Haowen Deng, Tolga Birdal, and Slobodan Ilic. Ppf-foldnet: Unsupervised learning of rotation invariant 3d local descriptors. In *Proceedings of the European conference on computer vision (ECCV)*, pages 602–618, 2018.

[54] Maximilian Denninger, Martin Sundermeyer, Dominik Winkelbauer, Youssef Zidan, Dmitry Olefir, Mohamad Elbadrawy, Ahsan Lodhi, and Harinandan Katam. Blenderproc. *arXiv preprint arXiv:1911.01911*, 2019.

[55] Maximilian Denninger, Dominik Winkelbauer, Martin Sundermeyer, Wout Boerdijk, Markus Knauer, Klaus H. Strobl, Matthias Humt, and Rudolph Triebel. Blenderproc2: A procedural pipeline for photorealistic rendering. *Journal of Open Source Software*, 8(82):4901, 2023.

[56] Theo Deprelle, Thibault Groueix, Matthew Fisher, Vladimir Kim, Bryan Russell, and Mathieu Aubry. Learning elementary structures for 3d shape generation and matching. In *Advances in Neural Information Processing Systems*, pages 7433–7443, 2019.

[57] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

[58] Runyu Ding, Jihan Yang, Chuhui Xue, Wenqing Zhang, Song Bai, and Xiaojuan Qi. Language-driven open-vocabulary 3d scene understanding. *arXiv preprint arXiv:2211.16312*, 2022.

[59] Laura Downs, Anthony Francis, Nate Koenig, Brandon Kinman, Ryan Hickman, Krista Reymann, Thomas B McHugh, and Vincent Vanhoucke. Google scanned objects: A high-quality dataset of 3d scanned household items. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2553–2560. IEEE, 2022.

[60] Benjamin Eckart, Wentao Yuan, Chao Liu, and Jan Kautz. Self-supervised learning on 3d point clouds by learning discrete generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8248–8257, 2021.

[61] Ziya Erkoç, Fangchang Ma, Qi Shan, Matthias Nießner, and Angela Dai. Hyperdiffusion: Generating implicit neural fields with weight-space diffusion. *arXiv preprint arXiv:2303.17015*, 2023.

[62] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 605–613, 2017.

[63] Huan Fu, Rongfei Jia, Lin Gao, Mingming Gong, Binqiang Zhao, Steve Maybank, and Dacheng Tao. 3d-future: 3d furniture shape with texture. *International Journal of Computer Vision*, 129:3313–3337, 2021.

[64] Xiao Fu, Wei Yin, Mu Hu, Kaixuan Wang, Yuexin Ma, Ping Tan, Shaojie Shen, Dahua Lin, and Xiaoxiao Long. Geowizard: Unleashing the diffusion priors for 3d geometry estimation from a single image. *arXiv preprint arXiv:2403.12013*, 2024.

[65] Matheus Gadelha, Aruni RoyChowdhury, Gopal Sharma, Evangelos Kalogerakis, Liangliang Cao, Erik Learned-Miller, Rui Wang, and Subhransu Maji. Label-efficient learning on point clouds using approximate convex decompositions. In *European Conference on Computer Vision*, pages 473–491. Springer, 2020.

[66] Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit H Bermano, Gal Chechik, and Daniel Cohen-Or. An image is worth one word: Personalizing text-to-image generation using textual inversion. *arXiv preprint arXiv:2208.01618*, 2022.

[67] Jun Gao, Tianchang Shen, Zian Wang, Wenzheng Chen, Kangxue Yin, Daiqing Li, Or Litany, Zan Gojcic, and Sanja Fidler. Get3d: A generative model of high quality 3d textured shapes learned from images. *Advances In Neural Information Processing Systems*, 35:31841–31854, 2022.

[68] Lin Gao, Jie Yang, Tong Wu, Yu-Jie Yuan, Hongbo Fu, Yu-Kun Lai, and Hao Zhang. Sdm-net: Deep generative network for structured deformable mesh. *ACM Transactions on Graphics (TOG)*, 38(6):1–15, 2019.

[69] Ruiqi Gao*, Aleksander Holynski*, Philipp Henzler, Arthur Brussee, Ricardo Martin-Brualla, Pratul P. Srinivasan, Jonathan T. Barron, and Ben Poole*. Cat3d: Create anything in 3d with multi-view diffusion models. *arXiv*, 2024.

[70] Yizhao Gao, Nanyi Fei, Guangzhen Liu, Zhiwu Lu, Tao Xiang, and Songfang Huang. Contrastive prototype learning with augmented embeddings for few-shot learning. *arXiv preprint arXiv:2101.09499*, 2021.

[71] Kyle Genova, Forrester Cole, Daniel Vlasic, Aaron Sarna, William T Freeman, and Thomas Funkhouser. Learning shape templates with structured implicit functions. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7154–7164, 2019.

[72] Martin Gerdzhev, Ryan Razani, Ehsan Taghavi, and Liu Bingbing. Tornado-net: multiview total variation semantic segmentation with diamond inception module. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9543–9549. IEEE, 2021.

[73] Mukulika Ghosh, Nancy M Amato, Yanyan Lu, and Jyh-Ming Lien. Fast approximate convex decomposition using relative concavity. *Computer-Aided Design*, 45(2):494–504, 2013.

[74] Elmer G Gilbert, Daniel W Johnson, and S Sathiya Keerthi. A fast procedure for computing the distance between complex objects in three-dimensional space. *IEEE Journal on Robotics and Automation*, 4(2):193–203, 1988.

[75] Rohit Girdhar, David F Fouhey, Mikel Rodriguez, and Abhinav Gupta. Learning a predictable and generative vector representation for objects. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VI 14*, pages 484–499. Springer, 2016.

[76] Georgia Gkioxari, Jitendra Malik, and Justin Johnson. Mesh r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9785–9795, 2019.

[77] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. 3d-coded: 3d correspondences by deep deformation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 230–246, 2018.

[78] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. A papier-mâché approach to learning 3d surface generation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 216–224, 2018.

[79] Thibault Groueix, Matthew Fisher, Vladimir G. Kim, Bryan C. Russell, and Mathieu Aubry. Deep self-supervised cycle-consistent deformation for few-shot shape segmentation. 2019.

[80] Gaël Guennebaud, Marcel Germann, and Markus Gross. Dynamic sampling and rendering of algebraic point set surfaces. In *Computer Graphics Forum*, volume 27, pages 653–662. Wiley Online Library, 2008.

[81] Stéphane Guinard and Loic Landrieu. Weakly supervised segmentation-aided classification of urban scenes from 3d lidar point clouds. In *ISPRS Workshop 2017*, 2017.

[82] Yuan-Chen Guo, Ying-Tian Liu, Ruizhi Shao, Christian Laforte, Vikram Voleti, Guan Luo, Chia-Hao Chen, Zi-Xin Zou, Chen Wang, Yan-Pei Cao, and Song-Hai Zhang. threestudio: A unified framework for 3d content generation. https://github.com/threestudio-project/threestudio, 2023.

[83] Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5356–5364,

2019.

[84] Anchit Gupta, Wenhan Xiong, Yixin Nie, Ian Jones, and Barlas Oğuz. 3dgen: Triplane latent diffusion for textured mesh generation. *arXiv preprint arXiv:2303.05371*, 2023.

[85] Huy Ha and Shuran Song. Semantic abstraction: Open-world 3d scene understanding from 2d vision-language models. In *Conference on Robot Learning*, 2022.

[86] Songfang Han, Jiayuan Gu, Kaichun Mo, Li Yi, Siyu Hu, Xuejin Chen, and Hao Su. Compositionally generalizable 3d structure prediction. *arXiv preprint arXiv:2012.02493*, 2020.

[87] Richard I Hartley. In defense of the eight-point algorithm. *IEEE Transactions on pattern analysis and machine intelligence*, 19(6):580–593, 1997.

[88] Kaveh Hassani and Mike Haley. Unsupervised multi-task feature learning on point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8160–8171, 2019.

[89] Tong He, Dong Gong, Zhi Tian, and Chunhua Shen. Learning and memorizing representative prototypes for 3d point cloud semantic and instance segmentation. In *European Conference on Computer Vision*, pages 564–580. Springer, 2020.

[90] Zexin He and Tengfei Wang. Openlrm: Open-source large reconstruction models. https://github.com/3DTopia/OpenLRM, 2023.

[91] Deepti Hegde, Jeya Maria Jose Valanarasu, and Vishal M Patel. Clip goes 3d: Leveraging prompt tuning for language grounded 3d recognition. *arXiv preprint arXiv:2303.11313*, 2023.

[92] Philipp Henzler, Jeremy Reizenstein, Patrick Labatut, Roman Shapovalov, Tobias Ritschel, Andrea Vedaldi, and David Novotny. Unsupervised learning of 3d object categories from videos in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4700–4709, 2021.

[93] Georg Hess, Johan Jaxing, Elias Svensson, David Hagerman, Christoffer Petersson, and Lennart Svensson. Masked autoencoder for self-supervised pre-training on lidar point clouds. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 350–359, 2023.

[94] Georg Hess, Adam Tonderski, Christoffer Petersson, Lennart Svensson, and Kalle Åström. Lidarclip or: How i learned to talk to point clouds. *arXiv preprint arXiv:2212.06858*, 2022.

[95] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[96] Fangzhou Hong, Mingyuan Zhang, Liang Pan, Zhongang Cai, Lei Yang, and Ziwei Liu. Avatarclip: Zero-shot text-driven generation and animation of 3d avatars. *arXiv preprint arXiv:2205.08535*, 2022.

[97] 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.

[98] Berthold KP Horn and Brian G Schunck. Determining optical flow. *Artificial intelligence*, 17(1-3):185–203, 1981.

[99] Ji Hou, Angela Dai, and Matthias Nießner. 3d-sis: 3d semantic instance segmentation of rgb-d scans. In

Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 4421–4430, 2019.

[100] Ji Hou, Benjamin Graham, Matthias Nießner, and Saining Xie. Exploring data-efficient 3d scene understanding with contrastive scene contexts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15587–15597, 2021.

[101] 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.

[102] Hanzhe Hu, Zhizhuo Zhou, Varun Jampani, and Shubham Tulsiani. Mvd-fusion: Single-view 3d via depth-consistent multi-view generation. In *CVPR*, 2024.

[103] Qingyong Hu, Bo Yang, Guangchi Fang, Yulan Guo, Ales Leonardis, Niki Trigoni, and Andrew Markham. Sqn: Weakly-supervised semantic segmentation of large-scale 3d point clouds with 1000x fewer labels. *arXiv preprint arXiv:2104.04891*, 2021.

[104] Haibin Huang, Evangelos Kalogerakis, Siddhartha Chaudhuri, Duygu Ceylan, Vladimir G. Kim, and Ersin Yumer. Learning local shape descriptors from part correspondences with multiview convolutional networks. *ACM Transactions on Graphics*, 2017.

[105] Rui Huang, Xuran Pan, Henry Zheng, Haojun Jiang, Zhifeng Xie, Shiji Song, and Gao Huang. Joint representation learning for text and 3d point cloud. *arXiv preprint arXiv:2301.07584*, 2023.

[106] Tianyu Huang, Bowen Dong, Yunhan Yang, Xiaoshui Huang, Rynson WH Lau, Wanli Ouyang, and Wangmeng Zuo. Clip2point: Transfer clip to point cloud classification with image-depth pre-training. *arXiv preprint arXiv:2210.01055*, 2022.

[107] Wenlong Huang, Igor Mordatch, Pieter Abbeel, and Deepak Pathak. Generalization in dexterous manipulation via geometry-aware multi-task learning. *arXiv preprint arXiv:2111.03062*, 2021.

[108] Zixuan Huang, Stefan Stojanov, Anh Thai, Varun Jampani, and James M Rehg. Planes vs. chairs: Category-guided 3d shape learning without any 3d cues. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part I*, pages 727–744. Springer, 2022.

[109] Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. Openclip, July 2021. If you use this software, please cite it as below.

[110] Ajay Jain, Ben Mildenhall, Jonathan T Barron, Pieter Abbeel, and Ben Poole. Zero-shot text-guided object generation with dream fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 867–876, 2022.

[111] Ajay Jain, Matthew Tancik, and Pieter Abbeel. Putting nerf on a diet: Semantically consistent few-shot view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5885–5894, 2021.

[112] Wonbong Jang and Lourdes Agapito. Codenerf: Disentangled neural radiance fields for object categories. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12949–12958, 2021.

[113] Krishna Murthy Jatavallabhula, Alihusein Kuwajerwala, Qiao Gu, Mohd Omama, Tao Chen, Shuang Li, Ganesh Iyer, Soroush Saryazdi, Nikhil Keetha, Ayush Tewari, et al. Conceptfusion: Open-set multimodal 3d

mapping. *arXiv preprint arXiv:2302.07241*, 2023.

[114] Nikolay Jetchev. Clipmatrix: Text-controlled creation of 3d textured meshes. *arXiv preprint arXiv:2109.12922*, 2021.

[115] 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.

[116] Hanwen Jiang, Zhenyu Jiang, Kristen Grauman, and Yuke Zhu. Few-view object reconstruction with unknown categories and camera poses. *arXiv preprint arXiv:2212.04492*, 2022.

[117] Hanwen Jiang, Zhenyu Jiang, Yue Zhao, and Qixing Huang. Leap: Liberate sparse-view 3d modeling from camera poses. *arXiv preprint arXiv:2310.01410*, 2023.

[118] Li Jiang, Hengshuang Zhao, Shaoshuai Shi, Shu Liu, Chi-Wing Fu, and Jiaya Jia. Pointgroup: Dual-set point grouping for 3d instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and Pattern recognition*, pages 4867–4876, 2020.

[119] Mohammad Mahdi Johari, Yann Lepoittevin, and François Fleuret. Geonerf: Generalizing nerf with geometry priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18365–18375, 2022.

[120] Andrew E Johnson. Spin-images: a representation for 3-d surface matching. 1997.

[121] Heewoo Jun and Alex Nichol. Shap-e: Generating conditional 3d implicit functions. *arXiv preprint arXiv:2305.02463*, 2023.

[122] Yannis Kalantidis, Mert Bulent Sariyildiz, Noe Pion, Philippe Weinzaepfel, and Diane Larlus. Hard negative mixing for contrastive learning. *Advances in Neural Information Processing Systems*, 33:21798–21809, 2020.

[123] Angjoo Kanazawa, Shubham Tulsiani, Alexei A Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 371–386, 2018.

[124] Animesh Karnewar, Andrea Vedaldi, David Novotny, and Niloy J Mitra. Holodiffusion: Training a 3d diffusion model using 2d images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18423–18433, 2023.

[125] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 7, 2006.

[126] Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. *ACM Transactions on Graphics (ToG)*, 32(3):1–13, 2013.

[127] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), July 2023.

[128] Justin Kerr, Chung Min Kim, Ken Goldberg, Angjoo Kanazawa, and Matthew Tancik. Lerf: Language embedded radiance fields. *arXiv preprint arXiv:2303.09553*, 2023.

[129] Nasir Khalid, Tianhao Xie, Eugene Belilovsky, and Tiberiu Popa. Text to mesh without 3d supervision using limit subdivision. *arXiv preprint arXiv:2203.13333*, 2022.

[130] Mijeong Kim, Seonguk Seo, and Bohyung Han. Infonerf: Ray entropy minimization for few-shot neural volume rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12912–12921, 2022.

[131] 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. *arXiv preprint arXiv:2304.02643*, 2023.

[132] Amit Pal Singh Kohli, Vincent Sitzmann, and Gordon Wetzstein. Semantic implicit neural scene representations with semi-supervised training. In *2020 International Conference on 3D Vision (3DV)*, pages 423–433. IEEE, 2020.

[133] Xin Kong, Shikun Liu, Xiaoyang Lyu, Marwan Taher, Xiaojuan Qi, and Andrew J Davison. Eschernet: A generative model for scalable view synthesis. *arXiv preprint arXiv:2402.03908*, 2024.

[134] Juil Koo, Ian Huang, Panos Achlioptas, Leonidas J Guibas, and Minhyuk Sung. Partglot: Learning shape part segmentation from language reference games. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16505–16514, 2022.

[135] Jonáš Kulhánek, Erik Derner, Torsten Sattler, and Robert Babuška. Viewformer: Nerf-free neural rendering from few images using transformers. In *European Conference on Computer Vision*, pages 198–216. Springer, 2022.

[136] Samuli Laine, Janne Hellsten, Tero Karras, Yeongho Seol, Jaakko Lehtinen, and Timo Aila. Modular primitives for high-performance differentiable rendering. *ACM Transactions on Graphics*, 39(6), 2020.

[137] Loic Landrieu and Guillaume Obozinski. Cut pursuit: Fast algorithms to learn piecewise constant functions on general weighted graphs. *SIAM Journal on Imaging Sciences*, 10(4):1724–1766, 2017.

[138] Loic Landrieu and Martin Simonovsky. Large-scale point cloud semantic segmentation with superpoint graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4558–4567, 2018.

[139] Han-Hung Lee and Angel X Chang. Understanding pure clip guidance for voxel grid nerf models. *arXiv preprint arXiv:2209.15172*, 2022.

[140] Jiachen Li, Quan Vuong, Shuang Liu, Minghua Liu, Kamil Ciosek, Henrik Christensen, and Hao Su. Multi-task batch reinforcement learning with metric learning. *Advances in neural information processing systems*, 33:6197–6210, 2020.

[141] Jiahao Li, Hao Tan, Kai Zhang, Zexiang Xu, Fujun Luan, Yinghao Xu, Yicong Hong, Kalyan Sunkavalli, Greg Shakhnarovich, and Sai Bi. Instant3d: Fast text-to-3d with sparse-view generation and large reconstruction model. *arXiv preprint arXiv:2311.06214*, 2023.

[142] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International Conference on Machine Learning*, pages 12888–12900. PMLR, 2022.

[143] Junnan Li, Pan Zhou, Caiming Xiong, and Steven CH Hoi. Prototypical contrastive learning of unsupervised

representations. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

[144] 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.

[145] Muheng Li, Yueqi Duan, Jie Zhou, and Jiwen Lu. Diffusion-sdf: Text-to-shape via voxelized diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12642–12651, 2023.

[146] Weiyu Li, Rui Chen, Xuelin Chen, and Ping Tan. Sweetdreamer: Aligning geometric priors in 2d diffusion for consistent text-to-3d. *arxiv:2310.02596*, 2023.

[147] Xuanlin Li, Yunhao Fang, Minghua Liu, Zhan Ling, Zhuowen Tu, and Hao Su. Distilling large vision-language model with out-of-distribution generalizability. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2492–2503, 2023.

[148] Victor Weixin Liang, Yuhui Zhang, Yongchan Kwon, Serena Yeung, and James Y Zou. Mind the gap: Understanding the modality gap in multi-modal contrastive representation learning. *Advances in Neural Information Processing Systems*, 35:17612–17625, 2022.

[149] Yiyi Liao, Simon Donne, and Andreas Geiger. Deep marching cubes: Learning explicit surface representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2916–2925, 2018.

[150] Jyh-Ming Lien and Nancy M Amato. Approximate convex decomposition. In *Proceedings of the twentieth annual symposium on Computational geometry*, pages 457–458, 2004.

[151] Jyh-Ming Lien and Nancy M Amato. Approximate convex decomposition of polyhedra. In *Proceedings of the 2007 ACM symposium on Solid and physical modeling*, pages 121–131, 2007.

[152] Jyh-Ming Lien and Nancy M Amato. Approximate convex decomposition of polyhedra and its applications. *Computer Aided Geometric Design*, 25(7):503–522, 2008.

[153] Amy Lin, Jason Y Zhang, Deva Ramanan, and Shubham Tulsiani. Relpose++: Recovering 6d poses from sparse-view observations. *arXiv preprint arXiv:2305.04926*, 2023.

[154] Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. Magic3d: High-resolution text-to-3d content creation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 300–309, 2023.

[155] Venice Erin Liong, Thi Ngoc Tho Nguyen, Sergi Widjaja, Dhananjai Sharma, and Zhuang Jie Chong. Amvnet: Assertion-based multi-view fusion network for lidar semantic segmentation. *arXiv preprint arXiv:2012.04934*, 2020.

[156] Guilin Liu, Zhonghua Xi, and Jyh-Ming Lien. Nearly convex segmentation of polyhedra through convex ridge separation. *Computer-Aided Design*, 78:137–146, 2016.

[157] Jinxian Liu, Minghui Yu, Bingbing Ni, and Ye Chen. Self-prediction for joint instance and semantic segmentation of point clouds. In *European Conference on Computer Vision*, pages 187–204. Springer, 2020.

[158] Minghua Liu, Xuanlin Li, Zhan Ling, Yangyan Li, and Hao Su. Frame mining: a free lunch for learning robotic manipulation from 3d point clouds. *arXiv preprint arXiv:2210.07442*, 2022.

[159] Minghua Liu, Hang Ma, Jiaoyang Li, and Sven Koenig. Task and path planning for multi-agent pickup and delivery. In *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2019.

[160] Minghua Liu, Lu Sheng, Sheng Yang, Jing Shao, and Shi-Min Hu. Morphing and sampling network for dense point cloud completion. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 11596–11603, 2020.

[161] 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. *arXiv preprint arXiv:2311.07885*, 2023.

[162] Minghua Liu, Ruoxi Shi, Kaiming Kuang, Yinhao Zhu, Xuanlin Li, Shizhong Han, Hong Cai, Fatih Porikli, and Hao Su. Openshape: Scaling up 3d shape representation towards open-world understanding. *Advances in Neural Information Processing Systems*, 36, 2024.

[163] Minghua Liu, Minhyuk Sung, Radomir Mech, and Hao Su. Deepmetahandles: Learning deformation meta-handles of 3d meshes with biharmonic coordinates. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12–21, 2021.

[164] 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, 2024.

[165] Minghua Liu, Chong Zeng, Xinyue Wei, Ruoxi Shi, Linghao Chen, Chao Xu, Mengqi Zhang, Zhaoning Wang, Xiaoshuai Zhang, Isabella Liu, et al. Meshformer: High-quality mesh generation with 3d-guided reconstruction model. *arXiv preprint arXiv:2408.10198*, 2024.

[166] Minghua Liu, Xiaoshuai Zhang, and Hao Su. Meshing point clouds with predicted intrinsic-extrinsic ratio guidance. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VIII 16*, pages 68–84. Springer, 2020.

[167] Minghua Liu, Yin Zhou, Charles R Qi, Boqing Gong, Hao Su, and Dragomir Anguelov. Less: Label-efficient semantic segmentation for lidar point clouds. In *European Conference on Computer Vision*, pages 70–89. Springer, 2022.

[168] 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.

[169] Rong Liu, Hao Zhang, and James Busby. Convex hull covering of polygonal scenes for accurate collision detection in games. In *Graphics Interface*, pages 203–210, 2008.

[170] 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.

[171] Shichen Liu, Tianye Li, Weikai Chen, and Hao Li. Soft rasterizer: A differentiable renderer for image-based 3d reasoning. 2019.

[172] 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.

[173] Yan Liu, Qingyong Hu, Yinjie Lei, Kai Xu, Jonathan Li, and Yulan Guo. Box2seg: Learning semantics of 3d point clouds with box-level supervision. *arXiv preprint arXiv:2201.02963*, 2022.

[174] Yongcheng Liu, Bin Fan, Shiming Xiang, and Chunhong Pan. Relation-shape convolutional neural network for point cloud analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8895–8904, 2019.

[175] Yuan Liu, Cheng Lin, Zijiao Zeng, Xiaoxiao Long, Lingjie Liu, Taku Komura, and Wenping Wang. Syncdreamer: Generating multiview-consistent images from a single-view image. *arXiv preprint arXiv:2309.03453*, 2023.

[176] Yuan Liu, Sida Peng, Lingjie Liu, Qianqian Wang, Peng Wang, Christian Theobalt, Xiaowei Zhou, and Wenping Wang. Neural rays for occlusion-aware image-based rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7824–7833, 2022.

[177] Yuxin Liu, Minshan Xie, Hanyuan Liu, and Tien-Tsin Wong. Text-guided texturing by synchronized multi-view diffusion. *arXiv preprint arXiv:2311.12891*, 2023.

[178] Zhen Liu, Yao Feng, Michael J Black, Derek Nowrouzezahrai, Liam Paull, and Weiyang Liu. Meshdiffusion: Score-based generative 3d mesh modeling. *arXiv preprint arXiv:2303.08133*, 2023.

[179] Zhengzhe Liu, Peng Dai, Ruihui Li, Xiaojuan Qi, and Chi-Wing Fu. Iss: Image as stetting stone for text-guided 3d shape generation. *arXiv preprint arXiv:2209.04145*, 2022.

[180] Zhengzhe Liu, Peng Dai, Ruihui Li, Xiaojuan Qi, and Chi-Wing Fu. Iss++: Image as stepping stone for text-guided 3d shape generation. *arXiv preprint arXiv:2303.15181*, 2023.

[181] Zhengzhe Liu, Xiaojuan Qi, and Chi-Wing Fu. One thing one click: A self-training approach for weakly supervised 3d semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1726–1736, 2021.

[182] Ziyuan Liu, Wei Liu, Yuzhe Qin, Fanbo Xiang, Minghao Gou, Songyan Xin, Maximo A Roa, Berk Calli, Hao Su, Yu Sun, et al. Ocrtoc: A cloud-based competition and benchmark for robotic grasping and manipulation. *IEEE Robotics and Automation Letters*, 7(1):486–493, 2021.

[183] Xiaoxiao Long, Yuan-Chen Guo, Cheng Lin, Yuan Liu, Zhiyang Dou, Lingjie Liu, Yuexin Ma, Song-Hai Zhang, Marc Habermann, Christian Theobalt, et al. Wonder3d: Single image to 3d using cross-domain diffusion. *arXiv preprint arXiv:2310.15008*, 2023.

[184] 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.

[185] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM siggraph computer graphics*, 21(4):163–169, 1987.

[186] J. Lorraine, K. Xie, X. Zeng, C. Lin, T. Takikawa, N. Sharp, T. Lin, M. Liu, S. Fidler, and J. Lucas. Att3d: Amortized text-to-3d object synthesis. In *2023 IEEE/CVF International Conference on Computer Vision*

(ICCV), pages 17900–17910, Los Alamitos, CA, USA, oct 2023. IEEE Computer Society.

[187] Yuanxun Lu, Jingyang Zhang, Shiwei Li, Tian Fang, David McKinnon, Yanghai Tsin, Long Quan, Xun Cao, and Yao Yao. Direct2.5: Diverse text-to-3d generation via multi-view 2.5d diffusion, 2024.

[188] Yuheng Lu, Chenfeng Xu, Xiaobao Wei, Xiaodong Xie, Masayoshi Tomizuka, Kurt Keutzer, and Shanghang Zhang. Open-vocabulary point-cloud object detection without 3d annotation. *arXiv preprint arXiv:2304.00788*, 2023.

[189] Huan Luo, Cheng Wang, Chenglu Wen, Ziyi Chen, Dawei Zai, Yongtao Yu, and Jonathan Li. Semantic labeling of mobile lidar point clouds via active learning and higher order mrf. *IEEE Transactions on Geoscience and Remote Sensing*, 56(7):3631–3644, 2018.

[190] Tiange Luo, Kaichun Mo, Zhiao Huang, Jiarui Xu, Siyu Hu, Liwei Wang, and Hao Su. Learning to group: A bottom-up framework for 3d part discovery in unseen categories. *arXiv preprint arXiv:2002.06478*, 2020.

[191] Xu Ma, Can Qin, Haoxuan You, Haoxi Ran, and Yun Fu. Rethinking network design and local geometry in point cloud: A simple residual mlp framework. *arXiv preprint arXiv:2202.07123*, 2022.

[192] Khaled Mamou and Faouzi Ghorbel. A simple and efficient approach for 3d mesh approximate convex decomposition. In *2009 16th IEEE international conference on image processing (ICIP)*, pages 3501–3504. IEEE, 2009.

[193] Khaled Mamou, E Lengyel, and AK Peters. Volumetric hierarchical approximate convex decomposition. In *Game Engine Gems 3*, pages 141–158. AK Peters, 2016.

[194] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 922–928. IEEE, 2015.

[195] Oier Mees, Maxim Tatarchenko, Thomas Brox, and Wolfram Burgard. Self-supervised 3d shape and viewpoint estimation from single images for robotics. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6083–6089. IEEE, 2019.

[196] Luke Melas-Kyriazi, Iro Laina, Christian Rupprecht, and Andrea Vedaldi. Realfusion: 360deg reconstruction of any object from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8446–8455, 2023.

[197] Luke Melas-Kyriazi, Christian Rupprecht, and Andrea Vedaldi. Pc2: Projection-conditioned point cloud diffusion for single-image 3d reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12923–12932, 2023.

[198] Benedikt Mersch, Xieyuanli Chen, Jens Behley, and Cyrill Stachniss. Self-supervised point cloud prediction using 3d spatio-temporal convolutional networks. In *Conference on Robot Learning*, pages 1444–1454. PMLR, 2022.

[199] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4460–4470, 2019.

[200] Gal Metzer, Elad Richardson, Or Patashnik, Raja Giryes, and Daniel Cohen-Or. Latent-nerf for shape-guided generation of 3d shapes and textures. In *Proceedings of the IEEE/CVF Conference on Computer Vision and*

Pattern Recognition, pages 12663–12673, 2023.

[201] Oscar Michel, Roi Bar-On, Richard Liu, Sagie Benaim, and Rana Hanocka. Text2mesh: Text-driven neural stylization for meshes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13492–13502, June 2022.

[202] 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.

[203] Brian Mirtich. V-clip: Fast and robust polyhedral collision detection. *ACM Transactions On Graphics (TOG)*, 17(3):177–208, 1998.

[204] Paritosh Mittal, Yen-Chi Cheng, Maneesh Singh, and Shubham Tulsiani. Autosdf: Shape priors for 3d completion, reconstruction and generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 306–315, 2022.

[205] Kaichun Mo, Paul Guerrero, Li Yi, Hao Su, Peter Wonka, Niloy Mitra, and Leonidas J Guibas. Structurenet: Hierarchical graph networks for 3d shape generation. *arXiv preprint arXiv:1908.00575*, 2019.

[206] 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.

[207] Nasir Mohammad Khalid, Tianhao Xie, Eugene Belilovsky, and Tiberiu Popa. Clip-mesh: Generating textured meshes from text using pretrained image-text models. In *SIGGRAPH Asia 2022 conference papers*, pages 1–8, 2022.

[208] Ron Mokady, Amir Hertz, and Amit H Bermano. Clipcap: Clip prefix for image captioning. *arXiv preprint arXiv:2111.09734*, 2021.

[209] Tongzhou Mu, Zhan Ling, Fanbo Xiang, Derek Yang, Xuanlin Li, Stone Tao, Zhiao Huang, Zhiwei Jia, and Hao Su. Maniskill: Generalizable manipulation skill benchmark with large-scale demonstrations. *arXiv preprint arXiv:2107.14483*, 2021.

[210] Tongzhou Mu, Minghua Liu, and Hao Su. Drs: Learning reusable dense rewards for multi-stage tasks. *arXiv preprint arXiv:2404.16779*, 2024.

[211] Norman Müller, Andrea Simonelli, Lorenzo Porzi, Samuel Rota Bulò, Matthias Nießner, and Peter Kontschieder. Autorf: Learning 3d object radiance fields from single view observations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3971–3980, 2022.

[212] Muhammad Ferjad Naeem, Evin Pınar Örnek, Yongqin Xian, Luc Van Gool, and Federico Tombari. 3d compositional zero-shot learning with decompositional consensus. In *European Conference on Computer Vision*, pages 713–730. Springer, 2022.

[213] Charlie Nash, Yaroslav Ganin, SM Ali Eslami, and Peter Battaglia. Polygen: An autoregressive generative model of 3d meshes. In *International conference on machine learning*, pages 7220–7229. PMLR, 2020.

[214] Diego Nehab, Szymon Rusinkiewicz, James Davis, and Ravi Ramamoorthi. Efficiently combining positions and normals for precise 3d geometry. *ACM Trans. Graph.*, 24(3):536–543, jul 2005.

[215] Alex Nichol, Heewoo Jun, Prafulla Dhariwal, Pamela Mishkin, and Mark Chen. Point-e: A system for

generating 3d point clouds from complex prompts. *arXiv preprint arXiv:2212.08751*, 2022.

[216] Alexandr Notchenko, Vladislav Ishimtsev, Alexey Artemov, Vadim Selyutin, Emil Bogomolov, and Evgeny Burnaev. Scan2part: Fine-grained and hierarchical part-level understanding of real-world 3d scans. *arXiv preprint arXiv:2206.02366*, 2022.

[217] OpenAI. Gpt-4 technical report, 2023.

[218] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2023.

[219] Joseph O'Rourke and Kenneth Supowit. Some np-hard polygon decomposition problems. *IEEE Transactions on Information Theory*, 29(2):181–190, 1983.

[220] Junyi Pan, Xiaoguang Han, Weikai Chen, Jiapeng Tang, and Kui Jia. Deep mesh reconstruction from single rgb images via topology modification networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9964–9973, 2019.

[221] Yatian Pang, Wenxiao Wang, Francis EH Tay, Wei Liu, Yonghong Tian, and Li Yuan. Masked autoencoders for point cloud self-supervised learning. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part II*, pages 604–621. Springer, 2022.

[222] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 165–174, 2019.

[223] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed AA Osman, Dimitrios Tzionas, and Michael J Black. Expressive body capture: 3d hands, face, and body from a single image. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10975–10985, 2019.

[224] Songyou Peng, Kyle Genova, Chiyu Jiang, Andrea Tagliasacchi, Marc Pollefeys, Thomas Funkhouser, et al. Openscene: 3d scene understanding with open vocabularies. *arXiv preprint arXiv:2211.15654*, 2022.

[225] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. In *ICLR*, 2023.

[226] Omid Poursaeed, Tianxing Jiang, Han Qiao, Nayun Xu, and Vladimir G Kim. Self-supervised learning of point clouds via orientation estimation. In *2020 International Conference on 3D Vision (3DV)*, pages 1018–1028. IEEE, 2020.

[227] Charles R Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. Frustum pointnets for 3d object detection from rgb-d data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 918–927, 2018.

[228] 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.

[229] 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.

[230] Zekun Qi, Runpei Dong, Guofan Fan, Zheng Ge, Xiangyu Zhang, Kaisheng Ma, and Li Yi. Contrast with reconstruct: Contrastive 3d representation learning guided by generative pretraining. *arXiv preprint arXiv:2302.02318*, 2023.

[231] Guocheng Qian, Yuchen Li, Houwen Peng, Jinjie Mai, Hasan Hammoud, Mohamed Elhoseiny, and Bernard Ghanem. Pointnext: Revisiting pointnet++ with improved training and scaling strategies. *arXiv:2206.04670*, 2022.

[232] Guocheng Qian, Jinjie Mai, Abdullah Hamdi, Jian Ren, Aliaksandr Siarohin, Bing Li, Hsin-Ying Lee, Ivan Skorokhodov, Peter Wonka, Sergey Tulyakov, and Bernard Ghanem. Magic123: One image to high-quality 3d object generation using both 2d and 3d diffusion priors. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024.

[233] Lingteng Qiu, Guanying Chen, Xiaodong Gu, Qi Zuo, Mutian Xu, Yushuang Wu, Weihao Yuan, Zilong Dong, Liefeng Bo, and Xiaoguang Han. Richdreamer: A generalizable normal-depth diffusion model for detail richness in text-to-3d, 2023.

[234] 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.

[235] Amit Raj, Srinivas Kaza, Ben Poole, Michael Niemeyer, Ben Mildenhall, Nataniel Ruiz, Shiran Zada, Kfir Aberman, Michael Rubenstein, Jonathan Barron, Yuanzhen Li, and Varun Jampani. Dreambooth3d: Subject-driven text-to-3d generation. *ICCV*, 2023.

[236] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

[237] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pages 8821–8831. PMLR, 2021.

[238] Yongming Rao, Jiwen Lu, and Jie Zhou. Global-local bidirectional reasoning for unsupervised representation learning of 3d point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5376–5385, 2020.

[239] Yongming Rao, Wenliang Zhao, Guangyi Chen, Yansong Tang, Zheng Zhu, Guan Huang, Jie Zhou, and Jiwen Lu. Denseclip: Language-guided dense prediction with context-aware prompting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18082–18091, 2022.

[240] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3d deep learning with pytorch3d. *arXiv preprint arXiv:2007.08501*, 2020.

[241] Jeremy Reizenstein, Roman Shapovalov, Philipp Henzler, Luca Sbordone, Patrick Labatut, and David Novotny. Common objects in 3d: Large-scale learning and evaluation of real-life 3d category reconstruction. In *International Conference on Computer Vision*, 2021.

[242] Konstantinos Rematas, Ricardo Martin-Brualla, and Vittorio Ferrari. Sharf: Shape-conditioned radiance fields from a single view. *arXiv preprint arXiv:2102.08860*, 2021.

[243] Yufan Ren, Tong Zhang, Marc Pollefeys, Sabine Süsstrunk, and Fangjinhua Wang. Volrecon: Volume rendering of signed ray distance functions for generalizable multi-view reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16685–16695, 2023.

[244] Zhongzheng Ren, Ishan Misra, Alexander G Schwing, and Rohit Girdhar. 3d spatial recognition without spatially labeled 3d. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13204–13213, 2021.

[245] Elad Richardson, Gal Metzer, Yuval Alaluf, Raja Giryes, and Daniel Cohen-Or. Texture: Text-guided texturing of 3d shapes. *arXiv preprint arXiv:2302.01721*, 2023.

[246] Joshua Robinson, Ching-Yao Chuang, Suvrit Sra, and Stefanie Jegelka. Contrastive learning with hard negative samples. *arXiv preprint arXiv:2010.04592*, 2020.

[247] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10674–10685, Los Alamitos, CA, USA, jun 2022. IEEE Computer Society.

[248] David Rozenberszki, Or Litany, and Angela Dai. Language-grounded indoor 3d semantic segmentation in the wild. *arXiv preprint arXiv:2204.07761*, 2022.

[249] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in Neural Information Processing Systems*, 35:36479–36494, 2022.

[250] 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.

[251] Aditya Sanghi. Info3d: Representation learning on 3d objects using mutual information maximization and contrastive learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIX 16*, pages 626–642. Springer, 2020.

[252] Aditya Sanghi, Hang Chu, Joseph G Lambourne, Ye Wang, Chin-Yi Cheng, Marco Fumero, and Kamal Rahimi Malekshan. Clip-forge: Towards zero-shot text-to-shape generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18603–18613, 2022.

[253] Jonathan Sauder and Bjarne Sievers. Self-supervised deep learning on point clouds by reconstructing space. *Advances in Neural Information Processing Systems*, 32, 2019.

[254] Scott Schaefer and Joe Warren. Dual marching cubes: Primal contouring of dual grids. In *12th Pacific Conference on Computer Graphics and Applications, 2004. PG 2004. Proceedings.*, pages 70–76. IEEE, 2004.

[255] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016.

[256] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *arXiv preprint arXiv:2210.08402*, 2022.

[257] Junyoung Seo, Wooseok Jang, Min-Seop Kwak, Jaehoon Ko, Hyeonsu Kim, Junho Kim, Jin-Hwa Kim, Jiyoung Lee, and Seungryong Kim. Let 2d diffusion model know 3d-consistency for robust text-to-3d generation. *arXiv preprint arXiv:2303.07937*, 2023.

[258] Charu Sharma and Manohar Kaul. Self-supervised few-shot learning on point clouds. *Advances in Neural Information Processing Systems*, 33:7212–7221, 2020.

[259] Gopal Sharma, Bidya Dash, Aruni RoyChowdhury, Matheus Gadelha, Marios Loizou, L Cao, Rui Wang, EG Learned-Miller, Subhransu Maji, and Evangelos Kalogerakis. Prifit: Learning to fit primitives improves few shot point cloud segmentation. In *Computer Graphics Forum*, volume 41, pages 39–50. Wiley Online Library, 2022.

[260] Gopal Sharma, Rishabh Goyal, Difan Liu, Evangelos Kalogerakis, and Subhransu Maji. Csgnet: Neural shape parser for constructive solid geometry. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5515–5523, 2018.

[261] Gopal Sharma, Kangxue Yin, Subhransu Maji, Evangelos Kalogerakis, Or Litany, and Sanja Fidler. Mvdecor: Multi-view dense correspondence learning for fine-grained 3d segmentation. *arXiv preprint arXiv:2208.08580*, 2022.

[262] Ruoxi Shi, Hansheng Chen, Zhuoyang Zhang, Minghua Liu, Chao Xu, Xinyue Wei, Linghao Chen, Chong Zeng, and Hao Su. Zero123++: a single image to consistent multi-view diffusion base model. *arXiv preprint arXiv:2310.15110*, 2023.

[263] Xian Shi, Xun Xu, Ke Chen, Lile Cai, Chuan Sheng Foo, and Kui Jia. Label-efficient point cloud semantic segmentation: An active learning approach. *arXiv preprint arXiv:2101.06931*, 2021.

[264] Yichun Shi, Peng Wang, Jianglong Ye, Mai Long, Kejie Li, and Xiao Yang. Mvdream: Multi-view diffusion for 3d generation. *arXiv:2308.16512*, 2023.

[265] Yawar Siddiqui, Antonio Alliegro, Alexey Artemov, Tatiana Tommasi, Daniele Sirigatti, Vladislav Rosov, Angela Dai, and Matthias Nießner. Meshgpt: Generating triangle meshes with decoder-only transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19615–19625, 2024.

[266] Samarth Sinha, Jason Y Zhang, Andrea Tagliasacchi, Igor Gilitschenski, and David B Lindell. Sparsepose: Sparse-view camera pose regression and refinement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21349–21359, 2023.

[267] Jake Snell, Kevin Swersky, and Richard S Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[268] Jack Snoeyink. Point location. In *Handbook of discrete and computational geometry*, pages 1005–1028. Chapman and Hall/CRC, 2017.

[269] Chao Sun, Zhedong Zheng, Xiaohan Wang, Mingliang Xu, and Yi Yang. Point cloud pre-training by mixing and disentangling. *arXiv e-prints*, pages arXiv–2109, 2021.

[270] Chun-Yu Sun, Yu-Qi Yang, Hao-Xiang Guo, Peng-Shuai Wang, Xin Tong, Yang Liu, and Heung-Yeung Shum. Semi-supervised 3d shape segmentation with multilevel consistency and part substitution. *arXiv preprint arXiv:2204.08824*, 2022.

[271] Jingxiang Sun, Bo Zhang, Ruizhi Shao, Lizhen Wang, Wen Liu, Zhenda Xie, and Yebin Liu. Dreamcraft3d: Hierarchical 3d generation with bootstrapped diffusion prior, 2023.

[272] Minhyuk Sung, Zhenyu Jiang, Panos Achlioptas, Niloy J. Mitra, and Leonidas J. Guibas. DeformSyncNet: Deformation transfer via synchronized shape deformation spaces, 2020.

[273] Qingyang Tan, Lin Gao, Yu-Kun Lai, and Shihong Xia. Meshvae: Variational autoencoders for deforming 3d mesh models. 2018.

[274] Qingyang Tan, Lin Gao, Yu-Kun Lai, Jie Yang, and Shihong Xia. Mesh-based autoencoders for localized deformation component analysis. 2018.

[275] Haotian Tang, Zhijian Liu, Shengyu Zhao, Yujun Lin, Ji Lin, Hanrui Wang, and Song Han. Searching efficient 3d architectures with sparse point-voxel convolution. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 685–702. Springer, 2020.

[276] Haotian Tang, Shang Yang, Zhijian Liu, Ke Hong, Zhongming Yu, Xiuyu Li, Guohao Dai, Yu Wang, and Song Han. Torchsparse++: Efficient training and inference framework for sparse convolution on gpus. In *IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2023.

[277] Jiaxiang Tang, Zhaoxi Chen, Xiaokang Chen, Tengfei Wang, Gang Zeng, and Ziwei Liu. Lgm: Large multi-view gaussian model for high-resolution 3d content creation. *arXiv preprint arXiv:2402.05054*, 2024.

[278] Jiaxiang Tang, Jiawei Ren, Hang Zhou, Ziwei Liu, and Gang Zeng. Dreamgaussian: Generative gaussian splatting for efficient 3d content creation. *arXiv preprint arXiv:2309.16653*, 2023.

[279] Junshu Tang, Tengfei Wang, Bo Zhang, Ting Zhang, Ran Yi, Lizhuang Ma, and Dong Chen. Make-it-3d: High-fidelity 3d creation from a single image with diffusion prior. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 22819–22829, October 2023.

[280] Ayush Tewari, Tianwei Yin, George Cazenavette, Semon Rezchikov, Joshua B Tenenbaum, Frédo Durand, William T Freeman, and Vincent Sitzmann. Diffusion with forward models: Solving stochastic inverse problems without direct supervision. *arXiv preprint arXiv:2306.11719*, 2023.

[281] Ali Thabet, Humam Alwassel, and Bernard Ghanem. Mortonnet: Self-supervised learning of local features in 3d point clouds. *arXiv preprint arXiv:1904.00230*, 2019.

[282] Ali Thabet, Humam Alwassel, and Bernard Ghanem. Self-supervised learning of local features in 3d point clouds. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 938–939, 2020.

[283] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6411–6420, 2019.

[284] Jesse Thomason, Mohit Shridhar, Yonatan Bisk, Chris Paxton, and Luke Zettlemoyer. Language grounding with 3d objects. In *Conference on Robot Learning*, pages 1691–1701. PMLR, 2022.

[285] Daniel Thul, L'ubor Ladickỳ, Sohyeon Jeong, and Marc Pollefeys. Approximate convex decomposition and transfer for animated meshes. *ACM Transactions on Graphics (TOG)*, 37(6):1–10, 2018.

[286] 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, 2024.

[287] Alex Trevithick and Bo Yang. Grf: Learning a general radiance field for 3d representation and rendering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15182–15192, 2021.

[288] 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.

[289] Vishaal Udandarao. Understanding and fixing the modality gap in vision-language models.

[290] Mikaela Angelina Uy, Quang-Hieu Pham, Binh-Son Hua, Thanh Nguyen, and Sai-Kit Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1588–1597, 2019.

[291] Mukund Varma, Peihao Wang, Xuxi Chen, Tianlong Chen, Subhashini Venugopalan, and Zhangyang Wang. Is attention all that nerf needs? In *The Eleventh International Conference on Learning Representations*, 2022.

[292] Vikram Voleti, Chun-Han Yao, Mark Boss, Adam Letts, David Pankratz, Dmitrii Tochilkin, Christian Laforte, Robin Rombach, and Varun Jampani. SV3D: Novel multi-view synthesis and 3D generation from a single image using latent video diffusion. *arXiv*, 2024.

[293] Thang Vu, Kookhoi Kim, Tung M Luu, Thanh Nguyen, and Chang D Yoo. Softgroup for 3d instance segmentation on point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2708–2717, 2022.

[294] Can Wang, Menglei Chai, Mingming He, Dongdong Chen, and Jing Liao. Clip-nerf: Text-and-image driven manipulation of neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3835–3844, 2022.

[295] Haiyan Wang, Xuejian Rong, Liang Yang, Jinglun Feng, Jizhong Xiao, and Yingli Tian. Weakly supervised semantic segmentation in 3d graph-structured point clouds of wild scenes. *arXiv preprint arXiv:2004.12498*, 2020.

[296] Hanchen Wang, Qi Liu, Xiangyu Yue, Joan Lasenby, and Matt J Kusner. Unsupervised point cloud pre-training via occlusion completion. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9782–9792, 2021.

[297] Haochen Wang, Xiaodan Du, Jiahao Li, Raymond A Yeh, and Greg Shakhnarovich. Score jacobian chaining: Lifting pretrained 2d diffusion models for 3d generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12619–12629, 2023.

[298] He Wang, Srinath Sridhar, Jingwei Huang, Julien Valentin, Shuran Song, and Leonidas J Guibas. Normalized object coordinate space for category-level 6d object pose and size estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2642–2651, 2019.

[299] Lingjing Wang, Xiang Li, and Yi Fang. Few-shot learning of part-specific probability space for 3d shape segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4504–4513, 2020.

[300] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *Proceedings of the European conference on computer vision (ECCV)*, pages 52–67, 2018.

[301] Peihao Wang, Xuxi Chen, Tianlong Chen, Subhashini Venugopalan, Zhangyang Wang, et al. Is attention all nerf needs? *arXiv preprint arXiv:2207.13298*, 2022.

[302] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *arXiv preprint arXiv:2106.10689*, 2021.

[303] Peng Wang and Yichun Shi. Imagedream: Image-prompt multi-view diffusion for 3d generation. *arXiv preprint arXiv:2312.02201*, 2023.

[304] Peng Wang, Hao Tan, Sai Bi, Yinghao Xu, Fujun Luan, Kalyan Sunkavalli, Wenping Wang, Zexiang Xu, and Kai Zhang. Pf-lrm: Pose-free large reconstruction model for joint pose and shape prediction. *arXiv preprint arXiv:2311.12024*, 2023.

[305] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul P Srinivasan, Howard Zhou, Jonathan T Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. Ibrnet: Learning multi-view image-based rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4690–4699, 2021.

[306] Ruocheng Wang, Yunzhi Zhang, Jiayuan Mao, Ran Zhang, Chin-Yi Cheng, and Jiajun Wu. Ikea-manual: Seeing shape assembly step by step. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022.

[307] Weiyue Wang, Duygu Ceylan, Radomir Mech, and Ulrich Neumann. 3dn: 3d deformation network. 2019.

[308] Weiyue Wang, Ronald Yu, Qiangui Huang, and Ulrich Neumann. Sgpn: Similarity group proposal network for 3d point cloud instance segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2569–2578, 2018.

[309] Xiaogang Wang, Xun Sun, Xinyu Cao, Kai Xu, and Bin Zhou. Learning fine-grained segmentation of 3d shapes without part labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10276–10285, 2021.

[310] Xinlong Wang, Shu Liu, Xiaoyong Shen, Chunhua Shen, and Jiaya Jia. Associatively segmenting instances and semantics in point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4096–4105, 2019.

[311] Yu Wang, Alec Jacobson, Jernej Barbič, and Ladislav Kavan. Linear subspace design for real-time shape deformation. 2015.

[312] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5):1–12, 2019.

[313] Zhen Wang, Qiangeng Xu, Feitong Tan, Menglei Chai, Shichen Liu, Rohit Pandey, Sean Fanello, Achuta Kadambi, and Yinda Zhang. Mvdd: Multi-view depth diffusion models, 2023.

[314] Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. *arXiv preprint*

arXiv:2305.16213, 2023.

[315] 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. *arXiv preprint arXiv:2403.05034*, 2024.

[316] Jiacheng Wei, Guosheng Lin, Kim-Hui Yap, Tzu-Yi Hung, and Lihua Xie. Multi-path region mining for weakly supervised 3d semantic segmentation on point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4384–4393, 2020.

[317] Jiacheng Wei, Hao Wang, Jiashi Feng, Guosheng Lin, and Kim-Hui Yap. Taps3d: Text-guided 3d textured shape generation from pseudo supervision, 2023.

[318] Xinyue Wei, Minghua Liu, Zhan Ling, and Hao Su. Approximate convex decomposition for 3d meshes with collision-aware concavity and tree search. *arXiv preprint arXiv:2205.02961*, 2022.

[319] Xinyue Wei, Fanbo Xiang, Sai Bi, Anpei Chen, Kalyan Sunkavalli, Zexiang Xu, and Hao Su. NeuManifold: Neural Watertight Manifold Reconstruction with Efficient and High-Quality Rendering Support. *arXiv preprint*, 2023.

[320] Xinyue Wei, Kai Zhang, Sai Bi, Hao Tan, Fujun Luan, Valentin Deschaintre, Kalyan Sunkavalli, Hao Su, and Zexiang Xu. Meshlrm: Large reconstruction model for high-quality mesh. *arXiv preprint arXiv:2404.12385*, 2024.

[321] Jean-Baptiste Weibel, Timothy Patten, and Markus Vincze. Sim2real 3d object classification using spherical kernel point convolution and a deep center voting scheme. *arXiv preprint arXiv:2103.06134*, 2021.

[322] Chao Wen, Yinda Zhang, Zhuwen Li, and Yanwei Fu. Pixel2mesh++: Multi-view 3d mesh generation via deformation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1042–1051, 2019.

[323] Haohan Weng, Tianyu Yang, Jianan Wang, Yu Li, Tong Zhang, CL Chen, and Lei Zhang. Consistent123: Improve consistency for one image to 3d object synthesis. *arXiv preprint arXiv:2310.08092*, 2023.

[324] Francis Williams, Teseo Schneider, Claudio Silva, Denis Zorin, Joan Bruna, and Daniele Panozzo. Deep geometric prior for surface reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10130–10139, 2019.

[325] Sangmin Woo, Byeongjun Park, Hyojun Go, Jin-Young Kim, and Changick Kim. Harmonyview: Harmonizing consistency and diversity in one-image-to-3d, 2023.

[326] Robert J. Woodham. *Photometric method for determining surface orientation from multiple images*, page 513–531. MIT Press, Cambridge, MA, USA, 1989.

[327] Chao-Yuan Wu, Justin Johnson, Jitendra Malik, Christoph Feichtenhofer, and Georgia Gkioxari. Multiview compressive coding for 3d reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9065–9075, 2023.

[328] Chin-Hsuan Wu, Yen-Chun Chen, Bolivar Solarte, Lu Yuan, and Min Sun. ifusion: Inverting diffusion for pose-free reconstruction from sparse views, 2023.

[329] Jiajun Wu, Yifan Wang, Tianfan Xue, Xingyuan Sun, Bill Freeman, and Josh Tenenbaum. Marrnet: 3d shape

reconstruction via 2.5 d sketches. *Advances in neural information processing systems*, 30, 2017.

[330] Tong Wu, Jiarui Zhang, Xiao Fu, Yuxin Wang, Jiawei Ren, Liang Pan, Wayne Wu, Lei Yang, Jiaqi Wang, Chen Qian, et al. Omniobject3d: Large-vocabulary 3d object dataset for realistic perception, reconstruction and generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 803–814, 2023.

[331] Tsung-Han Wu, Yueh-Cheng Liu, Yu-Kai Huang, Hsin-Ying Lee, Hung-Ting Su, Ping-Chia Huang, and Winston H Hsu. Redal: Region-based and diversity-aware active learning for point cloud semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 15510–15519, 2021.

[332] Yueh-Hua Wu, Jiashun Wang, and Xiaolong Wang. Learning generalizable dexterous manipulation from human grasp affordance. *arXiv preprint arXiv:2204.02320*, 2022.

[333] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015.

[334] Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, et al. Sapien: A simulated part-based interactive environment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11097–11107, 2020.

[335] Haozhe Xie, Hongxun Yao, Xiaoshuai Sun, Shangchen Zhou, and Shengping Zhang. Pix2vox: Context-aware 3d reconstruction from single and multi-view images. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2690–2698, 2019.

[336] Haozhe Xie, Hongxun Yao, Shengping Zhang, Shangchen Zhou, and Wenxiu Sun. Pix2vox++: Multi-scale context-aware 3d object reconstruction from single and multiple images. *International Journal of Computer Vision*, 128(12):2919–2935, 2020.

[337] Kevin Xie, Jonathan Lorraine, Tianshi Cao, Jun Gao, James Lucas, Antonio Torralba, Sanja Fidler, and Xiaohui Zeng. Latte3d: Large-scale amortized text-to-enhanced3d synthesis. *arXiv preprint arXiv:2403.15385*, 2024.

[338] Saining Xie, Jiatao Gu, Demi Guo, Charles R Qi, Leonidas Guibas, and Or Litany. Pointcontrast: Unsupervised pre-training for 3d point cloud understanding. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pages 574–591. Springer, 2020.

[339] 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. *arXiv preprint arXiv:2408.10195*, 2024.

[340] Dejia Xu, Yifan Jiang, Peihao Wang, Zhiwen Fan, Yi Wang, and Zhangyang Wang. Neurallift-360: Lifting an in-the-wild 2d photo to a 3d object with 360° views. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4479–4489, 2023.

[341] Jiale Xu, Weihao Cheng, Yiming Gao, Xintao Wang, Shenghua Gao, and Ying Shan. Instantmesh: Efficient 3d mesh generation from a single image with sparse-view large reconstruction models. *arXiv preprint arXiv:2404.07191*, 2024.

[342] Jiale Xu, Xintao Wang, Weihao Cheng, Yan-Pei Cao, Ying Shan, Xiaohu Qie, and Shenghua Gao. Dream3d: Zero-shot text-to-3d synthesis using 3d shape prior and text-to-image diffusion models. In *Proceedings of*

the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 20908–20918, 2023.

[343] Jianyun Xu, Ruixiang Zhang, Jian Dou, Yushi Zhu, Jie Sun, and Shiliang Pu. Rpvnet: A deep and efficient range-point-voxel fusion network for lidar point cloud segmentation. *arXiv preprint arXiv:2103.12978*, 2021.

[344] Katie Xu, Yasuhiro Yao, Kazuhiko Murasaki, Shingo Ando, and Atsushi Sagata. Semantic segmentation of sparsely annotated 3d point clouds by pseudo-labelling. In *Proceedings of the International Conference on 3D Vision (3DV)*, pages 463–471. IEEE, 2019.

[345] Qiangeng Xu, Weiyue Wang, Duygu Ceylan, Radomir Mech, and Ulrich Neumann. Disn: Deep implicit surface network for high-quality single-view 3d reconstruction. *Advances in neural information processing systems*, 32, 2019.

[346] Xianghao Xu, Yifan Ruan, Srinath Sridhar, and Daniel Ritchie. Unsupervised kinematic motion detection for part-segmented 3d shape collections. In *ACM SIGGRAPH 2022 Conference Proceedings*, pages 1–9, 2022.

[347] Xun Xu and Gim Hee Lee. Weakly supervised semantic point cloud segmentation: Towards 10x fewer labels. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13706–13715, 2020.

[348] Yinghao Xu, Zifan Shi, Wang Yifan, Sida Peng, Ceyuan Yang, Yujun Shen, and Wetzstein Gordon. Grm: Large gaussian reconstruction model for efficient 3d reconstruction and generation. *arxiv: 2403.14621*, 2024.

[349] Yinghao Xu, Hao Tan, Fujun Luan, Sai Bi, Peng Wang, Jiahao Li, Zifan Shi, Kalyan Sunkavalli, Gordon Wetzstein, Zexiang Xu, et al. Dmv3d: Denoising multi-view diffusion using 3d large reconstruction model. *arXiv preprint arXiv:2311.09217*, 2023.

[350] Le Xue, Mingfei Gao, Chen Xing, Roberto Martín-Martín, Jiajun Wu, Caiming Xiong, Ran Xu, Juan Carlos Niebles, and Silvio Savarese. Ulip: Learning unified representation of language, image and point cloud for 3d understanding. *arXiv preprint arXiv:2212.05171*, 2022.

[351] Farid Yagubbayli, Yida Wang, Alessio Tonioni, and Federico Tombari. Legoformer: Transformers for block-by-block multi-view 3d reconstruction. *arXiv preprint arXiv:2106.12102*, 2021.

[352] Xu Yan, Jiantao Gao, Jie Li, Ruimao Zhang, Zhen Li, Rui Huang, and Shuguang Cui. Sparse single sweep lidar point cloud segmentation via learning contextual shape priors from scene completion. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2020.

[353] Bo Yang, Jianan Wang, Ronald Clark, Qingyong Hu, Sen Wang, Andrew Markham, and Niki Trigoni. Learning object bounding boxes for 3d instance segmentation on point clouds. *Advances in neural information processing systems*, 32, 2019.

[354] Cheng-Kun Yang, Ji-Jia Wu, Kai-Syun Chen, Yung-Yu Chuang, and Yen-Yu Lin. An mil-derived transformer for weakly supervised point cloud segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11830–11839, 2022.

[355] Daniel Yang, Tarik Tosun, Benjamin Eisner, Volkan Isler, and Daniel Lee. Robotic grasping through combined image-based grasp proposal and 3d reconstruction. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6350–6356. IEEE, 2021.

[356] Hao Yang, Lanqing Hong, Aoxue Li, Tianyang Hu, Zhenguo Li, Gim Hee Lee, and Liwei Wang. Contranerf: Generalizable neural radiance fields for synthetic-to-real novel view synthesis via contrastive learning. In

Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 16508–16517, 2023.

[357] Hong-Ming Yang, Xu-Yao Zhang, Fei Yin, and Cheng-Lin Liu. Robust classification with convolutional prototype learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3474–3482, 2018.

[358] Jihan Yang, Runyu Ding, Zhe Wang, and Xiaojuan Qi. Regionplc: Regional point-language contrastive learning for open-world 3d scene understanding. *arXiv preprint arXiv:2304.00962*, 2023.

[359] Sheng Yang, Kang Chen, Minghua Liu, Hongbo Fu, and Shi-Min Hu. Saliency-aware real-time volumetric fusion for object reconstruction. In *Computer Graphics Forum*, volume 36, pages 167–174. Wiley Online Library, 2017.

[360] Sheng Yang, Beichen Li, Minghua Liu, Yu-Kun Lai, Leif Kobbelt, and Shi-Min Hu. Heterofusion: Dense scene reconstruction integrating multi-sensors. *IEEE transactions on visualization and computer graphics*, 26(11):3217–3230, 2019.

[361] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Interpretable unsupervised learning on 3d point clouds. *arXiv preprint arXiv:1712.07262*, 2017.

[362] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 206–215, 2018.

[363] Jianglong Ye, Peng Wang, Kejie Li, Yichun Shi, and Heng Wang. Consistent-1-to-3: Consistent image to 3d view synthesis via geometry-aware diffusion models. *arXiv preprint arXiv:2310.03020*, 2023.

[364] 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.

[365] Li Yi, Wang Zhao, He Wang, Minhyuk Sung, and Leonidas J Guibas. Gspn: Generative shape proposal network for 3d instance segmentation in point cloud. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3947–3956, 2019.

[366] Wang Yifan, Noam Aigerman, Vladimir Kim, Siddhartha Chaudhuri, and Olga Sorkine-Hornung. Neural cages for detail-preserving 3d deformations. 2020.

[367] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4578–4587, 2021.

[368] Chaohui Yu, Qiang Zhou, Jingliang Li, Zhe Zhang, Zhibin Wang, and Fan Wang. Points-to-3d: Bridging the gap between sparse points and shape-controllable text-to-3d generation. In *Proceedings of the 31st ACM International Conference on Multimedia*, MM '23, page 6841–6850, New York, NY, USA, 2023. Association for Computing Machinery.

[369] Fenggen Yu, Kun Liu, Yan Zhang, Chenyang Zhu, and Kai Xu. Partnet: A recursive part decomposition network for fine-grained and hierarchical shape segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9491–9500, 2019.

[370] Wang Yu, Xuelin Qian, Jingyang Huo, Tiejun Huang, Bo Zhao, and Yanwei Fu. Pushing the limits of 3d shape generation at scale. *arXiv preprint arXiv:2306.11510*, 2023.

[371] Wangbo Yu, Li Yuan, Yan-Pei Cao, Xiangjun Gao, Xiaoyu Li, Wenbo Hu, Long Quan, Ying Shan, and Yonghong Tian. Hifi-123: Towards high-fidelity one image to 3d content generation, 2024.

[372] Xumin Yu, Lulu Tang, Yongming Rao, Tiejun Huang, Jie Zhou, and Jiwen Lu. Point-bert: Pre-training 3d point cloud transformers with masked point modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19313–19322, 2022.

[373] Xiaohui Zeng, Arash Vahdat, Francis Williams, Zan Gojcic, Or Litany, Sanja Fidler, and Karsten Kreis. Lion: Latent point diffusion models for 3d shape generation. *arXiv preprint arXiv:2210.06978*, 2022.

[374] Yihan Zeng, Chenhan Jiang, Jiageng Mao, Jianhua Han, Chaoqiang Ye, Qingqiu Huang, Dit-Yan Yeung, Zhen Yang, Xiaodan Liang, and Hang Xu. Clipˆ 2: Contrastive language-image-point pretraining from real-world point cloud data. *arXiv preprint arXiv:2303.12417*, 2023.

[375] Biao Zhang, Jiapeng Tang, Matthias Niessner, and Peter Wonka. 3dshape2vecset: A 3d shape representation for neural fields and generative diffusion models. *arXiv preprint arXiv:2301.11445*, 2023.

[376] Biao Zhang and Peter Wonka. Point cloud instance segmentation using probabilistic embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8883–8892, 2021.

[377] Haotian Zhang, Pengchuan Zhang, Xiaowei Hu, Yen-Chun Chen, Liunian Harold Li, Xiyang Dai, Lijuan Wang, Lu Yuan, Jenq-Neng Hwang, and Jianfeng Gao. Glipv2: Unifying localization and vision-language understanding. *arXiv preprint arXiv:2206.05836*, 2022.

[378] Jason Y Zhang, Deva Ramanan, and Shubham Tulsiani. Relpose: Predicting probabilistic relative rotation for single objects in the wild. In *European Conference on Computer Vision*, pages 592–611. Springer, 2022.

[379] Junbo Zhang, Runpei Dong, and Kaisheng Ma. Clip-fo3d: Learning free open-world 3d scene representations from 2d dense clip. *arXiv preprint arXiv:2303.04748*, 2023.

[380] Kai Zhang, Sai Bi, Hao Tan, Yuanbo Xiangli, Nanxuan Zhao, Kalyan Sunkavalli, and Zexiang Xu. Gs-lrm: Large reconstruction model for 3d gaussian splatting. *arXiv*, 2024.

[381] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3836–3847, 2023.

[382] Lyumin Zhang. Reference-only control. https://github.com/Mikubill/sd-webui-controlnet/discussions/1236, 2023.

[383] 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.

[384] Xiaoshuai Zhang, Sai Bi, Kalyan Sunkavalli, Hao Su, and Zexiang Xu. Nerfusion: Fusing radiance fields for large-scale scene reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5449–5458, 2022.

[385] Xiaoshuai Zhang, Rui Chen, Ang Li, Fanbo Xiang, Yuzhe Qin, Jiayuan Gu, Zhan Ling, Minghua Liu, Peiyu Zeng, Songfang Han, et al. Close the optical sensing domain gap by physics-grounded active stereo sensor simulation. *IEEE Transactions on Robotics*, 39(3):2429–2447, 2023.

[386] Yachao Zhang, Zonghao Li, Yuan Xie, Yanyun Qu, Cuihua Li, and Tao Mei. Weakly supervised semantic segmentation for large-scale point cloud. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 3421–3429, 2021.

[387] Yachao Zhang, Yanyun Qu, Yuan Xie, Zonghao Li, Shanshan Zheng, and Cuihua Li. Perturbed self-distillation: Weakly supervised large-scale point cloud semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15520–15528, 2021.

[388] Zaiwei Zhang, Rohit Girdhar, Armand Joulin, and Ishan Misra. Self-supervised pretraining of 3d features on any point-cloud. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10252–10263, 2021.

[389] Na Zhao, Tat-Seng Chua, and Gim Hee Lee. Few-shot 3d point cloud semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8873–8882, 2021.

[390] Zibo Zhao, Wen Liu, Xin Chen, Xianfang Zeng, Rui Wang, Pei Cheng, Bin Fu, Tao Chen, Gang Yu, and Shenghua Gao. Michelangelo: Conditional 3d shape generation based on shape-image-text aligned latent representation. *arXiv preprint arXiv:2306.17115*, 2023.

[391] Xin-Yang Zheng, Hao Pan, Yu-Xiao Guo, Xin Tong, and Yang Liu. Mvd^2: Efficient multiview 3d reconstruction for multiview diffusion. In *SIGGRAPH*, 2024.

[392] Xin-Yang Zheng, Hao Pan, Peng-Shuai Wang, Xin Tong, Yang Liu, and Heung-Yeung Shum. Locally attentional sdf diffusion for controllable 3d shape generation. *arXiv preprint arXiv:2305.04461*, 2023.

[393] Shuaifeng Zhi, Tristan Laidlow, Stefan Leutenegger, and Andrew J Davison. In-place scene labelling and understanding with implicit scene representation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15838–15847, 2021.

[394] Yin Zhou, Pei Sun, Yu Zhang, Dragomir Anguelov, Jiyang Gao, Tom Ouyang, James Guo, Jiquan Ngiam, and Vijay Vasudevan. End-to-end multi-view fusion for 3d object detection in lidar point clouds. In *Conference on Robot Learning*, pages 923–932. PMLR, 2020.

[395] Zhizhuo Zhou and Shubham Tulsiani. Sparsefusion: Distilling view-conditioned diffusion for 3d reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12588–12597, 2023.

[396] Xiangyang Zhu, Renrui Zhang, Bowei He, Ziyao Zeng, Shanghang Zhang, and Peng Gao. Pointclip v2: Adapting clip for powerful 3d open-world learning. *arXiv preprint arXiv:2211.11682*, 2022.

[397] Xinge Zhu, Hui Zhou, Tai Wang, Fangzhou Hong, Yuexin Ma, Wei Li, Hongsheng Li, and Dahua Lin. Cylindrical and asymmetrical 3d convolution networks for lidar segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9939–9948, 2021.

[398] Zi-Xin Zou, Zhipeng Yu, Yuan-Chen Guo, Yangguang Li, Ding Liang, Yan-Pei Cao, and Song-Hai Zhang. Triplane meets gaussian splatting: Fast and generalizable single-view 3d reconstruction with transformers, 2023.

[399] Silvia Zuffi, Angjoo Kanazawa, and Michael J Black. Lions and tigers and bears: Capturing non-rigid, 3d, articulated shape from images. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 3955–3963, 2018.

[400] Silvia Zuffi, Angjoo Kanazawa, David W Jacobs, and Michael J Black. 3d menagerie: Modeling the 3d shape and pose of animals. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6365–6373, 2017.