UNIVERSITY OF CALIFORNIA SAN DIEGO

Learning to Capture, Understand, and Generate Large-Scale 3D Scenes

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

in

Computer Science

by

Xiaoshuai Zhang

2024

University of California San Diego

2024

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I am deeply grateful to my PhD advisor, Professor Hao Su, for his unwavering guidance and support throughout my doctoral journey. His profound insights and dedication have not only shaped this dissertation but also significantly contributed to my growth as a researcher. Professor Su's commitment to pushing the boundaries of knowledge and his openness to exploring new ideas have inspired me to pursue academic excellence with courage and curiosity. I am truly thankful for the opportunity to learn from such a remarkable scholar.

In addition, I extend my heartfelt gratitude to Professor Leonidas Guibas, whose unparalleled knowledge and unique perspectives have greatly enriched my research during my time at Stanford and Google Research. His ability to connect contemporary challenges with traditional graphics and learning methods, even those from decades ago, has provided invaluable insights that have deepened my understanding of the field. Despite his demanding schedule, Professor Guibas has always made time for thoughtful discussions, and his wisdom has played a crucial role in shaping my research journey.

My sincere thanks go out to all my labmates who accompanied me throughout this painstaking yet rewarding journey, including (in alphabetical order): Rui Chen, Jiayuan Gu, Songfang Han, Zhiao Huang, Zhiwei Jia, Zhan Ling, Fangchen Liu, Isabella Liu, Yulin Liu, Tongzhou Mu, Yuzhe Qin, Ruoxi Shi, Fanbo Xiang, and Xiaodi Yuan. Special thanks are due to Minghua Liu and Xinyue Wei. Your camaraderie, encouragement, and willingness to share knowledge have made the challenges easier to overcome and the successes more meaningful. The countless discussions, late-night Zoom brainstorming sessions, and shared experiences have not only contributed to my growth as a researcher but also created memories I will cherish for a lifetime. I am truly grateful to have been part of such a supportive and inspiring community.

I also wish to express my profound gratitude to the brilliant minds I had the privilege of working with during my industry internships, where I found not only mentors

but also lifelong friends. To Jen-Hao Rick Chang, Ashish Shrivastava, Hema Swetha Koppula, and Oncel Tuzel at Apple, your guidance and support were invaluable. At Adobe, I am deeply thankful to Zexiang Xu, Sai Bi, and Kalyan Sunkavalli for your insights and collaboration. And at Google, my sincere thanks go to Kyle Genova, Zhicheng Wang, Howard Zhou, Abhijit Kundu, Soham Ghosh, Danushen Gnanapragasam, Varun Jampani, Thomas Funkhouser, Guandao Yang, Songyou Peng, and Mikaela Angelina Uy. Each of you has contributed to my growth in ways I will always appreciate, and the knowledge, inspiration, and friendships forged during our time together will remain with me throughout my career.

I feel incredibly fortunate to have collaborated with so many brilliant researchers, whose expertise and dedication have enriched this journey in countless ways. I would like to express my sincere thanks to each of you (listed alphabetically): Anpei Chen, Linghao Chen, Xi Chen, Xuejin Chen, Haian Jin, Ang Li, Leizhi Li, Dai Liu, Zhu Liu, Qing Ran, Jianyu Tao, Yijia Weng, Hongzhi Wu, Chao Xu, Jing Xu, Edward Yang, Kaizhi Yang, Jingyi Yu, Peiyu Zeng, Chong Zeng, Mengqi Zhang, Fuqiang Zhao, and Xiaowei Zhou. Collaborating with you has been a truly rewarding experience, and I have learned so much from each of you. Thank you for your insights, dedication, and the shared commitment to advancing our research.

My deepest gratitude goes to the cherished ones I have loved, liked, and fallen for along this journey. The PhD path has not been without its storms, and the same can be said for the beautiful, albeit sometimes bittersweet, moments shared with you. I treasure the unforgettable memories and precious time spent together, which have continually inspired me to grow into a better person. To my dear friends and those closest to my heart, Jiahao Ye, Zifeng Li, Ziwei Zheng, Zhuangze Huang, Xuezheng Hong, Zeyu Pan, and many more, your steadfast support and companionship have been the pillars that held me up during the most challenging times. I truly couldn't have come this far without you by my side. I hope I have brought you as much joy and support as you have given me,

and I look forward to continuing this journey together, staying connected, and creating even more lasting memories.

Lastly, but with the deepest reverence, I would like to express my profound gratitude to my parents and family. Your unconditional love has been the foundation upon which all my achievements rest. I am eternally grateful for the wisdom and life lessons you have imparted to me, shaping not only my mind but also my character. Your boundless trust and support have given me the freedom to pursue my dreams and make choices that have led me to where I am today. With your constant encouragement, I have grown into the person I am, surrounded by happiness and a deep sense of fulfillment. I am forever indebted to you for the love, strength, and guidance that have illuminated my path.

Chapter 2, in full, is a reprint of the material published in the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR): "NeRFusion: Fusing Radiance Fields for Large-Scale Scene Reconstruction" (Xiaoshuai Zhang, Sai Bi, Kalyan Sunkavalli, Hao Su, Zexiang Xu). The dissertation author was the primary investigator and author of this paper.

Chapter 3, in full, is a reprint of the material published in the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR): "Nerflets: Local Radiance Fields for Efficient Structure-Aware 3D Scene Representation from 2D Supervision" (Xiaoshuai Zhang, Abhijit Kundu, Thomas Funkhouser, Leonidas Guibas, Hao Su, Kyle Genova). The dissertation author was the primary investigator and author of this paper. We sincerely thank Yueyu Hu, Nilesh Kulkarni, Songyou Peng, Mikaela Angelina Uy, Boxin Wang, and Guandao Yang for useful discussion and help on experiments. We also thank Avneesh Sud for feedback on the manuscript.

Chapter 4, in full, is a reprint of the material published in the 2024 European Conference on Computer Vision (ECCV): "ConDense: Consistent 2D/3D Pre-training for Dense and Sparse Features from Multi-View Images” (Xiaoshuai Zhang, Zhicheng Wang, Howard Zhou, Soham Ghosh, Danushen Gnanapragasam, Varun Jampani, Hao Su, Leonidas Guibas). The dissertation author was the primary investigator and author of this paper. Our thanks go to Hao-Ning Wu and Bhav Ashok for their support in building large-scale pose estimation and NeRF pipeline for dataset pre-processing.

VITA

2019	Bachelor of Science, Peking University
2024	Master of Science, University of California San Diego
2024	Doctor of Philosophy, University of California San Diego

PUBLICATIONS

[1] Xiaoshuai Zhang, Zhicheng Wang, Howard Zhou, Soham Ghosh, Danushen Gnanapragasam, Varun Jampani, Hao Su, and Leonidas Guibas. "ConDense: Consistent 2D/3D Pre-training for Dense and Sparse Features from Multi-View Images", European Conference on Computer Vision (ECCV), 2024

[2] Kaizhi Yang, Xiaoshuai Zhang, Zhiao Huang, Xuejin Chen, Zexiang Xu, and Hao Su. "MovingParts: Motion-based 3D Part Discovery in Dynamic Radiance Field", International Conference on Learning Representations (ICLR), 2024

[3] 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

[4] Xiaoshuai Zhang*, Rui Chen*, Ang Li, 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 Optical Sensing Domain Gap by Physics-Grounded Active Stereo Sensor Simulation", IEEE Transactions on Robotics (T-RO), 2023, vol. 39, no. 3, pp. 2429-2447

[5] Rui Chen, Isabella Liu, Edward Yang, Jianyu Tao, Xiaoshuai Zhang, Qing Ran, Zhu Liu, Jing Xu, and Hao Su. "ActiveZero++: Mixed Domain Learning Stereo and Confidence-Based Depth Completion with Zero Annotation", IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 2023, vol. 45, no. 12, pp. 14098-14113

[6] Haian Jin, Isabella Liu, Peijia Xu, Xiaoshuai Zhang, Songfang Han, Sai Bi, Xiaowei Zhou, Zexiang Xu, and Hao Su. "TensoIR: Tensorial Inverse Rendering", Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2023, pp. 165-174

[7] Xiaoshuai Zhang, Abhijit Kundu, Thomas Funkhouser, Leonidas Guibas, Hao Su, and Kyle Genova. "Nerflets: Local Radiance Fields for Efficient Structure-Aware 3D Scene Representation from 2D Supervision", Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2023, pp. 8274-8284

[8] Jen-Hao Rick Chang, Ashish Shrivastava, Hema Swetha Koppula, Xiaoshuai Zhang, and Oncel Tuzel. "Style Equalization: Unsupervised Learning of Controllable Generative Sequence Models", International Conference on Machine Learning (ICML), 2022, pp. 2917-2937

[9] Xiaoshuai Zhang, Sai Bi, Kalyan Sunkavalli, Hao Su, and Zexiang Xu. “NeRFusion: Fusing Radiance Fields for Large-Scale Scene Reconstruction”, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022, pp. 5449-5458

[10] Isabella Liu, Edward Yang, Jianyu Tao, Rui Chen, Xiaoshuai Zhang, Qing Ran, Zhu Liu, and Hao Su. “ActiveZero: Mixed Domain Learning for Active Stereovision With Zero Annotation”, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022, pp. 13033-13042

[11] 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”, Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2021, pp. 14124-14133

[12] Minghua Liu, Xiaoshuai Zhang, and Hao Su. “Meshing Point Clouds with Predicted Intrinsic-Extrinsic Ratio Guidance”, European Conference on Computer Vision (ECCV), 2020, pp. 68-84

[13] Jiaying Liu, Dong Liu, Wenhan Yang, Sifeng Xia, Xiaoshuai Zhang, and Yuanying Dai. “A Comprehensive Benchmark for Single Image Compression Artifact Reduction”, IEEE Transactions on Image Processing (TIP), 2020, vol. 29, pp. 7845-7860

[14] Rosaura G. VidalMata, Sreya Banerjee, Brandon RichardWebster, Michael Albright, Pedro Davalos, Scott McCloskey, Ben Miller, Asong Tambo, Sushobhan Ghosh, Sudarshan Nagesh, Ye Yuan, Yueyu Hu, Junru Wu, Wenhan Yang, Xiaoshuai Zhang, Jiaying Liu, Zhangyang Wang, Hwann-Tzong Chen, Tzu-Wei Huang, Wen-Chi Chin, Yi-Chun Li, Mahmoud Lababidi, Charles Otto, and Walter J. Scheirer. “Bridging the Gap Between Computational Photography and Visual Recognition”, IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 2020, vol. 43, no. 12, pp. 4272-4284

[15] Shaofan Cai*, Xiaoshuai Zhang*, Haoqiang Fan, Haibin Huang, Jiangyu Liu, Jiaming Liu, Jiaying Liu, Jue Wang, and Jian Sun. “Disentangled Image Matting”, Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 8819-8828

[16] Xinlei Pan*, Weiyao Wang*, Xiaoshuai Zhang*, Bo Li, Jinfeng Yi, and Dawn Song. “How You Act Tells a Lot: Privacy-Leakage Attack on Deep Reinforcement Learning”, International Conference on Autonomous Agents and MultiAgent Systems (AAMAS), 2019, pp. 368-376

[17] Xiaoshuai Zhang*, Yiping Lu*, Jiaying Liu, and Bin Dong. “Dynamically Unfolding Recurrent Restorer: A Moving Endpoint Control Method for Image Restoration”, International Conference on Learning Representations (ICLR), 2019

[18] Xiaoshuai Zhang, Wenhan Yang, Yueyu Hu, and Jiaying Liu. “DMCNN: Dual-domain Multi-scale Convolutional Neural Network for Compression Artifacts Removal”, 2018 25th IEEE International Conference on Image Processing (ICIP), 2018, pp. 390-394

ABSTRACT OF THE DISSERTATION

Learning to Capture, Understand, and Generate Large-Scale 3D Scenes

by

Xiaoshuai Zhang

Doctor of Philosophy in Computer Science

University of California San Diego, 2024

Professor Hao Su, Chair

As the world becomes increasingly digitized, the demand for advanced 3D scene understanding has expanded beyond academic research into practical applications such as virtual reality (VR), augmented reality (AR), autonomous robotics, urban planning, and entertainment industries like gaming and film. The central aim of this dissertation is to push the boundaries of how we capture, interpret, and generate these large-scale 3D scenes, advancing both theoretical understanding and practical implementations.

Our key contributions include a novel framework, NeRFusion, for fast and scalable radiance field reconstruction, specifically designed for large indoor environments. By utilizing recurrent neural networks and sparse voxel grids, this framework achieves a balance between geometric accuracy and photorealism, significantly improving efficiency over traditional methods. Additionally, this dissertation introduces “nerflets”, an innovative 3D scene representation that breaks down complex scenes into smaller, interpretable radiance fields. This allows for more efficient storage and enhanced semantic understanding, enabling advanced tasks like 3D panoptic segmentation and interactive scene editing. The dissertation also proposes the ConDense pre-training scheme, which unifies 2D and 3D feature learning. Through a ray-marching process inspired by Neural Radiance Fields (NeRF), ConDense ensures consistent 2D-3D feature alignment during pre-training, improving performance across various downstream tasks, such as 2D and 3D classification and segmentation tasks, and cross-modality scene query and retrieval. Finally, the dissertation briefly touches on a novel methodology for generating 3D scenes by combining 2D diffusion models with 3D implicit scene representations, highlighting a promising direction for further study in the field.

The research pushes the boundaries of 3D scene capturing, understanding, and generation, offering solutions that are both practical and theoretically significant. These innovations not only advance the field but also provide valuable tools for industries reliant on high-fidelity 3D environments, paving the way for more intelligent, interactive digital worlds.

Chapter 1

Introduction

1.1 Learning to Capture, Understand, and Generate Large-Scale 3D Scenes

The ability to capture, understand, and generate large-scale 3D scenes is a critical capability at the intersection of computer vision, robotics, and computer graphics. As the world becomes increasingly digitized, the demand for advanced 3D scene understanding has expanded beyond academic research into practical applications such as virtual reality (VR), augmented reality (AR), autonomous robotics, urban planning, and entertainment industries like gaming and film. The central aim of this dissertation is to push the boundaries of how we capture, interpret, and generate these large-scale 3D scenes, advancing both theoretical understanding and practical implementations.

In recent years, 3D scene understanding has evolved from simple geometric reconstruction to more complex tasks that require a deep comprehension of the scene's semantics, dynamics, and interrelationships between objects. These advancements are crucial for applications that demand not just a static snapshot of an environment but a rich, interactive model that can be manipulated, explored, and analyzed in real-time. For instance, in autonomous driving, the ability of a vehicle to perceive its surroundings accurately, including recognizing and predicting the movement of objects within the scene, directly impacts its safety and efficiency. Similarly, in VR/AR, the generation of realistic and interactive environments hinges on the accurate reconstruction and understanding of 3D spaces.

The motivation behind this dissertation stems from the limitations of existing methods in dealing with large-scale 3D environments, which are often characterized by their complexity, diversity, and the need for real-time processing. Traditional methods, while effective in controlled settings or with limited scope, struggle with scalability, efficiency, and generalization. This work seeks to address these challenges by introducing novel methodologies that leverage the latest advances in deep learning, neural rendering, and multi-modal data fusion to improve the capture, understanding, and generation of large-scale 3D scenes.

1.2 Challenges in Large-Scale 3D Scene Capturing and Understanding

Understanding large-scale 3D scenes poses a multitude of challenges that arise from the inherent complexity of the environments, the vast amounts of data involved, and the need for high levels of accuracy and efficiency in processing. These challenges can be broadly categorized into three main areas: data acquisition and processing, representation and storage, and semantic understanding and interaction.

1.2.1 Data Acquisition and Processing

The first major challenge lies in the acquisition of 3D data, which must capture the intricate details of large and complex environments. Traditional methods rely on RGB-D cameras, LiDAR, or multi-view stereo techniques, each with its own limitations. RGB-D sensors, while capable of capturing depth information alongside color, are often limited in range and resolution, leading to incomplete or noisy data. LiDAR provides high-resolution point clouds but is expensive and generates enormous datasets that are challenging to process in real-time. Multi-view stereo techniques, on the other hand, require a large number of images and careful calibration, making them impractical for dynamic or large-scale scenes.

Moreover, processing this data to create a coherent 3D representation involves significant computational resources. Techniques such as volumetric fusion and point cloud processing need to handle the large volume of data efficiently while maintaining accuracy. Errors in data acquisition, such as occlusions or varying lighting conditions, further complicate the processing pipeline, often requiring sophisticated algorithms to fill in missing information or correct for inconsistencies.

1.2.2 Representation and Storage

Once the data is acquired, representing it in a form that is both compact and capable of supporting various downstream tasks is another significant challenge. Traditional representations like voxel grids or mesh models, while straightforward, suffer from scalability issues as the resolution of the scene increases. High-resolution voxel grids, for example, require enormous amounts of memory, making them impractical for large-scale scenes. Mesh models, although more efficient in terms of memory usage, struggle with representing complex geometries and dynamic scenes.

Recently, neural representations have emerged as a promising alternative. Methods such as Neural Radiance Fields (NeRF) have shown remarkable success in generating photorealistic images from sparse 2D views by encoding the scene within a neural network. However, these methods are computationally intensive and typically require optimization for each individual scene, limiting their applicability to real-time or large-scale scenarios. Furthermore, they often lack the ability to represent scene semantics, which is crucial for tasks such as object recognition, segmentation, and interaction.

1.2.3 Semantic Understanding and Interaction

Beyond geometric reconstruction, a deeper understanding of the scene's semantics is essential for enabling intelligent interaction with the environment. Semantic understanding involves recognizing and classifying objects within the scene, understanding their spatial relationships, and interpreting the scene's overall structure. This level of understanding is necessary for a wide range of applications, from autonomous navigation and manipulation in robotics to interactive editing and augmentation in AR/VR.

The challenge here is twofold. First, existing methods often require extensive labeled 3D datasets for training, which are expensive and time-consuming to acquire. Second, even with sufficient data, achieving real-time performance while maintaining high accuracy remains a significant hurdle. Many state-of-the-art methods are either too slow for practical use or lack the robustness needed to handle the diversity and variability present in real-world scenes.

1.3 Overview of Techniques and Contributions

This dissertation addresses these challenges by introducing several novel techniques designed to enhance the capture, understanding, and generation of large-scale 3D scenes. The following sections outline the key contributions of this work, each aimed at overcoming specific limitations in the current state of the art.

1.3.1 Fast and Scalable Radiance Field Reconstruction for Indoor Scenes

One of the central contributions of this dissertation is the development of a fast and scalable framework for radiance field reconstruction, specifically tailored for large-scale indoor scenes. Traditional reconstruction methods, such as Truncated Signed Distance Function (TSDF) fusion, focus primarily on geometry and are limited in their ability to produce realistic visualizations. On the other hand, neural rendering techniques like

NeRF offer photorealistic results but are computationally prohibitive and typically require per-scene optimization, making them unsuitable for large-scale or real-time applications.

This dissertation introduces a novel approach that leverages recurrent neural networks (RNNs) to incrementally build a sparse radiance field from a sequence of RGB images. By utilizing sparse voxel grids and tri-linearly interpolated neural features, the proposed method achieves a balance between geometric accuracy and photorealism, while significantly reducing the computational overhead. Unlike previous methods that optimize a global neural network for each scene, this approach generalizes across different scenes, making it more practical for applications that require quick adaptation to new environments.

1.3.2 Compact and Efficient 3D Scene Representation with Local Radiance Fields

Another major contribution is the introduction of "nerflets," a novel representation that offers a more compact and interpretable way to encode large-scale 3D scenes. Nerflets combine the strengths of local neural fields with a structured and irregular representation, enabling efficient scene decomposition into meaningful components. This method not only improves the efficiency of storage and manipulation but also enhances the interpretability of the scene, facilitating tasks such as 3D panoptic segmentation, novel view synthesis, and interactive editing.

Unlike traditional neural rendering approaches that rely on global MLP networks, nerflets allow for a localized and structured representation that can be jointly optimized from 2D images. This results in a comprehensive 3D decomposition that captures not just the appearance and geometry of the scene, but also its semantics and object instances. The compactness and efficiency of this representation make it particularly suitable for applications where real-time performance and scalability are crucial.

1.3.3 Large-Scale Consistent 2D-3D Pre-Training with Dense and Sparse Features

The third contribution of this dissertation is the development of ConDense, a novel pre-training scheme designed to address the challenge of 3D data scarcity. By leveraging large-scale 2D multi-view datasets and pre-trained 2D networks, ConDense enforces 2D-3D feature consistency through a ray-marching process inspired by NeRF. This process aligns features across the two domains, enabling the model to learn joint 2D-3D features that are consistent and transferrable.

The dual representation created by ConDense, which includes both dense per-pixel features and sparse keypoint-based features, allows for versatile and adaptable performance across a wide range of downstream tasks. This unified embedding space facilitates the efficient querying and matching of 2D images with large-scale 3D scenes, opening up new possibilities for cross-modal applications such as natural language-driven scene interrogation, image-based localization, and object retrieval.

1.3.4 Fast 3D Scene Generation by Lifting Panorama Images from 2D Diffusion Models

The generation of 3D scenes from 2D priors represents a transformative approach in the intersection of computer vision and graphics, aiming to leverage the strengths of both 2D diffusion models and 3D implicit scene representations. This methodology harnesses the power of 2D generative models to create detailed visual content, such as panoramas or walkthrough videos, and then reconstructs these 2D outputs into full 3D environments using advanced 3D reconstruction techniques.

At the heart of this approach lies the synergy between 2D and 3D representations. The process begins with a 2D diffusion model, which is conditioned on textual or image input to generate high-quality panorama images or video sequences that capture the essence of the desired scene. These 2D outputs, while rich in detail and aesthetics, lack the geometric depth required for full 3D scene understanding and interaction. To bridge this gap, we employ state-of-the-art 3D reconstruction methods including Neural Radiance Fields (NeRF) or 3D Gaussian Splatting (3DGS). These techniques take the 2D-generated content and infer a corresponding 3D structure, effectively converting the flat images into native 3D representations.

The integration of 2D diffusion models and 3D reconstruction algorithms offers several advantages. Firstly, it capitalizes on the maturity and robustness of 2D generative models, which have been extensively trained on vast image datasets, allowing them to produce visually compelling scenes. Secondly, by reconstructing these scenes in 3D, we unlock the potential for immersive experiences, enabling applications such as virtual reality (VR), augmented reality (AR), and 3D content creation.

This 2D-to-3D generation pipeline represents a novel contribution to the field, addressing challenges in scene synthesis by combining the creative flexibility of 2D models with the spatial accuracy of 3D techniques. This fusion not only enhances the quality and realism of generated 3D scenes but also opens new avenues for research and applications in areas requiring high-fidelity 3D content.

Chapter 2

Fast and Scalable Radiance Field Reconstruction for Indoor Scenes

2.1 Introduction

Reconstructing and rendering large-scale indoor scenes from RGB images is challenging but crucial for various applications in computer vision and graphics, including AR/VR, e-commerce, and robotics. While truncated signed distance function (TSDF) fusion techniques [120, 183] can achieve efficient reconstruction, these methods often use depth sensors and focus on geometric reconstruction only, and cannot synthesize realistic images. Recently, NeRF [115] proposed optimizing scene radiance fields, represented using global MLPs, from RGB images to achieve photo-realistic novel view synthesis. However, NeRF cannot handle large-scale scenes well due to its limited MLP network capacity and impractical slow per-scene optimization.

In this work, we aim to achieve *fast, large-scale* scene-level radiance field reconstruction to make neural scene reconstruction and rendering more practical. As opposed to small-scale object-centric scenes, we use “large-scale” to refer to full-size indoor scenes, like ScanNet scenes [36]), with multiple rooms and objects with complex scene geometry and appearance.

To achieve fast radiance field reconstruction on such challenging scenes, we propose a novel neural framework that uses recurrent neural modules to incrementally reconstruct

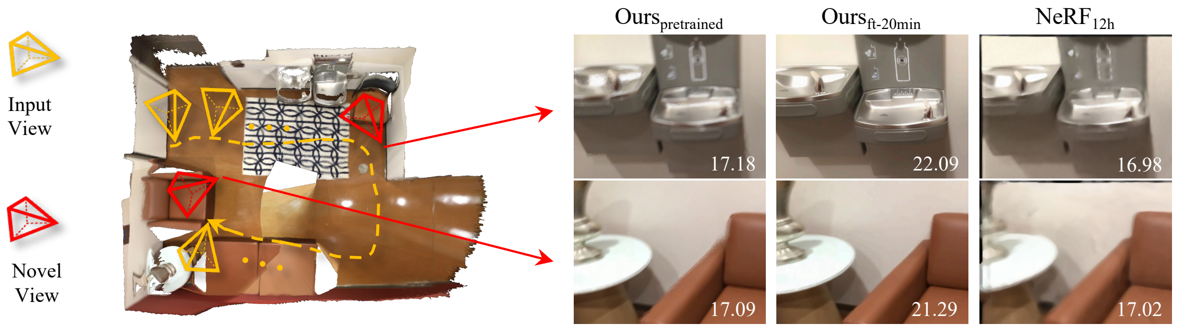

Figure 2.1. We propose a novel method to enable *fast* reconstruction of volumetric radiance fields of *large-scale* scenes. Our method uses a novel recurrent network – that is *generalizable* and trained across scenes – to sequentially reconstruct a radiance field of a large indoor scene in ScanNet [36] from an input image sequence (marked in yellow) via direct inference. The predicted field can be directly used to render realistic images at novel viewpoints (marked in red), achieving comparable quality to NeRF [115] that takes 12 hours per-scene optimization. This radiance field can be further fine-tuned for a short period of 20 min, leading to boosted quality that significantly outperforms NeRF.

a large sparse radiance field from a long RGB image sequence. Unlike NeRF [115], that requires per-scene optimization, our network is generalizable, pre-trained across scenes, and able to efficiently reconstruct large-scale radiance fields via direct network inference. As shown in Figure 2.1, our framework can successfully reconstruct a large indoor scene from from an input monocular RGB video from ScanNet [36], to create a high-quality radiance field with photo-realistic novel view synthesis results.

Our reconstructed radiance field is represented by a sparse volume grid with per-voxel neural features; these voxel features are tri-linearly interpolated at any scene location, and used to regress volume density and view-dependent radiance through an MLP decoder for differentiable volume rendering. In contrast to previous methods [98, 60] that reconstruct similar representations using slow per-scene optimization, we present a novel deep neural network that can be trained across scenes and generalize on unseen novel scenes to achieve fast radiance field reconstruction, bypassing per-scene fitting.

Given an input sequence of RGB images with known camera poses (that can be registered by SLAM or SfM techniques), our framework reconstructs a radiance field as a sparse neural volume. Our pipeline is inspired by the classical TSDF fusion workflow [120, 121, 183] that starts from per-view geometry (depth) and fuses the per-view reconstruction across key frames to obtain a global sparse TSDF volume. This workflow is widely used to reconstruct large-scale scenes, but only focuses on geometric reconstruction. Instead, we propose novel neural modules to reconstruct radiance fields as sparse voxels for photo-realistic rendering.

We first reconstruct local radiance fields for each input key frame. We leverage deep MVS techniques and apply sparse 3D convolutions on a world-space cost-volume built from unprojected 2D images features (regressed from a deep 2D CNN) of neighboring key frames. This reconstructs sparse neural voxels that represent a local radiance field. Once estimated, this field can already be used to render realistic images locally, though only for partial scene content seen by the local frames. We propose a recurrent neural fusion module to sequentially fuse multiple local fields across frames. Our fusion module recurrently takes a newly estimated local field as input and learns to incorporate the local voxels to progressively reconstruct a global radiance field modeling the entire scene, by adding new voxels and updating existing voxels. Our full model is trained from end to end, learning to reconstruct radiance fields with arbitrary scene scales from an arbitrary number of input images. We show that our direct network output can already render high-quality images; moreover, our neural field can be effectively fine-tuned by optimizing the predicted voxel features per scene in a short period to achieve better rendering quality (see Figure 2.1 and 2.4).

We train our full framework from end to end with only rendering losses on a combination of scenes from the ScanNet [36], DTU [72], and Google Scanned Object [143] datasets. These datasets contain a large variety of different objects and scenes, allowing for our method to work properly with any scene scale. We demonstrate, on various datasets, that our approach performs better than prior arts, including IBRNet [180] that also designs networks that generalize across scenes. Especially on large-scale indoor scenes, our results from real-time direct network inference can even be on par with NeRF's results from long per-scene optimization. Moreover, after only one hour of per-scene fine-tuning, our quality can be further boosted to the state-of-the-art, outperforming NeRF [113] and NVSF [98] that require much longer per-scene optimization times. Our approach significantly improves the efficiency and scalability of radiance field reconstruction. We believe this is an important step towards making neural scene reconstruction and rendering practical.

2.2 Related Work

Multi-view scene reconstruction. Abundant research has been conducted on reproducing the appearance of of 3D scens from multi-view data. To reconstruct the geometry, previous methods apply multi-view stereo [150, 152] or depth sensors [120] to acquire the depth information of the scene. Recently, learning-based multi-view stereo methods [69, 205, 206, 28, 110, 50, 32] based on plane-swept cost volumes are also introduced for depth estimation. Given the depth, one category of methods represent scenes with colored point clouds [3, 84, 89], and utilize point splatting to render images of the scene. Another category of methods [120, 121, 183] fuse multi-view depth and reconstruct surface meshes using techniques such as TSDF fusion or Poisson reconstruction, and further generate textures [233, 10] from multi-view images. However, both kinds of methods are sensitive to potential inaccuracies in point clouds and meshes resulting from corrupted depth, especially when there are thin structures and textureless regions, thus suffering from holes and blurry artifacts in the final renderings. While some works apply neural networks [3, 148, 59] such as a 2D CNN in screen-space to mitigate potential errors in the geometry, their models are per-scene optimized for a specific scene (similar to NeRF), requiring a long optimization time. Moreover, the screen-space neural networks typically produce temporally unstable results with flickering artifacts. Instead of estimating and fusing per-view depth, previous methods [75, 163, 15] introduce learning-based methods to aggregate per-view features and predict opacity volumes or signed distance volumes. These methods only focus on geometry reconstruction and cannot produce realistic renderings. In contrast, our approach models scenes as neural volumetric radiance fields and can reproduce the faithful scene appearance, producing photo-realistic novel views. Our pipeline is pretrained on multi-view image datasets and can generalize to novel scenes at arbitrary scales and enable efficient large-scale neural reconstruction.

Neural Radiance Fields. Volumetric representations [106, 115, 98] have been widely adopted to reconstruct the appearance of the scene. NeRF [115] uses a global MLP to regress the volume density and view-dependent radiance at any arbitrary point in the space, and applies volume rendering to synthesize images at novel viewpoints. Following works extend the framework for different tasks such as relighting [12, 11, 14, 160], scene editing [193] and dynamic scene modeling [127, 128, 93]. Similar to NeRF, most of these works train MLP networks, specific for each scene from scratch, which can take hours and even days to optimize, heavily time-consuming. On the other hand, the limited network capacity of MLPs makes these methods hard to scale up to large scene reconstruction. NVSF [98] improves the scalability by building sparse voxel grids with per-voxel features. However their networks and features are still optimized per scene from scratch and it can still take days for large-scale scenes. In contrast, while we use a similar sparse volume, our volume is generated from the direct inference of a pre-trained network, leading to fast large-scale scene reconstruction. The direct network output can be further fine-tuned in a short period to achieve better rendering quality than NeRF and NSVF.

Some previous papers also extend NeRF for generalization. PixelNeRF [208] uses 2D CNNs to extract image features on each sampled point of each ray used in ray marching for regressing the point's volume properties. However, their network is designed for object rendering with few images and is trained specifically for each dataset. IBRNet [180] uses a similar network but has better designs that enable rendering on any scene scales. However,

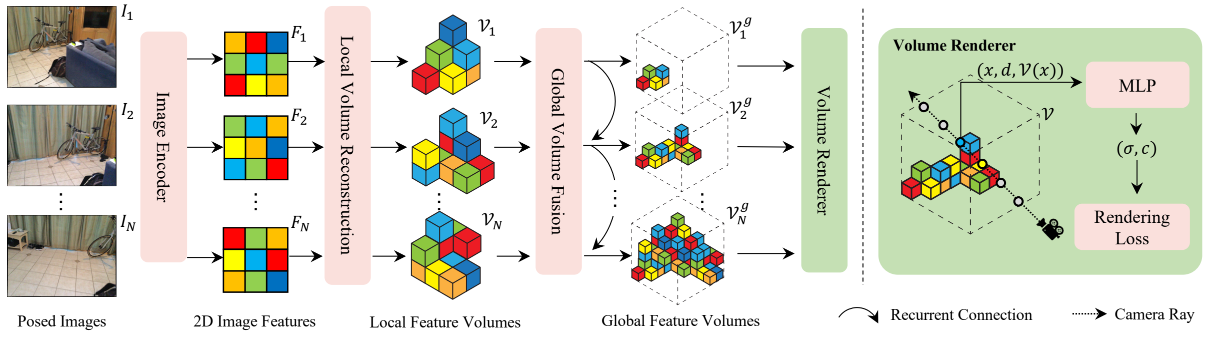

Figure 2.2. Overview of our framework. Given a sequence of images, 1) we first extract their image features ($F_1 \dots F_N$) using a 2D CNN. 2) Then, at each frame, we reconstruct a local sparse neural volume $\mathcal{V}_1 \dots \mathcal{V}_N$ in the canonical world space by fetching and aggregating 2D features across its neighboring views at visible voxels using a sparse 3D CNN. 3) We further fuse the local sparse volumes across frames using a recurrent neural network and sequentially build global feature volumes $\mathcal{V}_1^g \dots \mathcal{V}_N^g$ to model a radiance field of the entire scene. We regress volume density and view-dependent radiance from the sparse neural volumes to render images with differentiable ray marching.

it leverages image features from neighboring views as input, varying across novel viewpoints, which often lead to blurry or flickering artifacts from sparse inputs. Our network instead reconstructs a neural volume with per-voxel features in 3D, modeling scene geometry and appearance in a more consistent way, leading to much better rendering than IBRNet. MVSNeRF [23] also reconstructs 3D volumes; however, it focuses on reconstructing a local volume from a fixed number of three nearby views. In contrast, our network can fuse per-view local reconstruction into a global volume from an arbitrary number of images, leading to highly efficient large-scale scene reconstruction and rendering.

2.3 Method

We now present our approach for neural scene reconstruction and rendering. Given an input sequence of images I_1,...,I_N of a large-scale scene with their known camera parameters Φ_1,...,Φ_N, our approach reconstructs a radiance field modeling the entire scene for realistic rendering.

Our final output radiance field is represented by a sparse neural volume $\mathcal{V}^g$ with per-voxel neural features (Sec. 2.3.1). Unlike per-scene optimization methods [115, 98], we propose a deep neural network that sequentially takes the images I_t frame by frame as input and convert the sequence to the final sparse reconstruction via direct network inference. Our pipeline first learns to reconstruct a sparse volume $\mathcal{V}_t$ per input image frame, expressing a local radiance field covered by local frames (Sec. 2.3.2). We then leverage a recurrent fusion module that learns to fuse the per-frame volumes $\mathcal{V}_t$ online, incrementally reconstructing the global large-scale field $\mathcal{V}^g$ (Sec. 2.3.3). We train our full pipeline from end to end with pure rendering losses. Our model can reconstruct high-quality radiance fields from direct network inference; the estimated field can also be further fine-tuned to boost its quality (Sec. 2.3.4).

2.3.1 Sparse Volumes for Radiance Fields

Our output radiance field is modeled by a sparse neural volume $\mathcal{V}$ that has per-voxel neural features in voxels that approximately cover the actual scene surface. We regress volume density σ and view-dependent radiance c at any given 3D location x from this volume using an MLP network, in which we first tri-linearly sample a feature vector and then use the MLP to convert the feature to volume properties, expressed by

$$\sigma, c = R(x, d, \mathcal{V}(x)). \tag{2.1}$$

Here $\mathcal{V}(x)$ represents the trilinearly interpolated feature at x, R is the MLP, and d is the viewing direction in rendering. The output volume properties regressed from the volume can be directly used to synthesize images at novel target viewpoints via differentiable ray marching as is done in NeRF [115]. This radiance field representation is similar to the one in Neural Sparse Voxel Fields [98] that purely relies on per-scene optimization for the reconstruction. We instead propose to leverage neural networks trained across scenes to predict the neural volumes from image sequences.

In our pipeline, such sparse volumes are reconstructed locally as $\mathcal{V}_t$ per frame t and also globally as $\mathcal{V}_g$ for the entire sequence. The MLP network R is shared across all volumes in the training process. We model the local volumes and the global volume both in the canonical world space.

2.3.2 Reconstructing Local Volumes

We propose a deep neural network to regress a local neural volume for each input frame t, using its image I_t and $K-1$ images from neighboring views. Usually, given a monocular video, these neighboring views correspond to temporal neighboring frames. Using multiple nearby images for per-frame reconstruction allows the network to leverage multi-view correspondence to recover better scene geometry, which a single image cannot provide.

To make the local reconstruction per frame well generalized across scenes, we leverage deep MVS techniques [205, 74], which are known to be generalizable. We extract 2D image features, build a cost volume from the features, and regress a neural feature volume from the cost volume. However, unlike MVSNeRF [23] and other MVS techniques [205, 32] that built frustum volumes in view's perspective coordinate, we construct volumes in the canonical world coordinate frame to align it with the final global volume output $\mathcal{V}_g$, facilitating the following fusion process.

Image feature extraction. We use a deep 2D convolutional neural network to extract 2D image features for each input image. This network maps the input image I_t into a 2D feature map F_t, encoding the scene content from each view.

Local sparse volume. We consider the bounding box that covers the frustums of all K neighboring viewpoints in the world coordinate frame, containing of a set of voxels in the canonical space. The bounding volume is axis-aligned with the world frame; each voxel inside it can be visible to a different number of neighboring views. We mask out all the voxels invisible to all view, leading to a sparse set of voxels in the bounding box. We then unproject the image features into this volume for our local reconstruction.

3D feature volume. For each neighboring viewpoint i and its feature map F_i, we build a 3D feature volume $\mathcal{U}_i$. In particular, for each visible voxel centered at v, we fetch the 2D image feature at its 2D projection from each neighboring view at frame t. In addition to pure image features, we leverage the corresponding viewing direction d_i at v from each viewpoint and compute additional features using an MLP G. The per-view 3D volume $\mathcal{U}_i$ is expressed by

$$\mathcal{U}_i(v) = [F_i(u_i), G(d_i)], \tag{2.2}$$

where $\mathcal{U}_i(v)$ is the feature at a voxel centered at v, u_i is the center's 2D projection in view i, $[\cdot, \cdot]$ represents feature concatenation. Note that, we encode the additional information of input viewing directions in the reconstruction process; this crucial information makes our following fusion module effectively account for the view-dependent effects captured across frames.

Neural reconstruction. We then aggregate the features across multiple neighboring viewpoints to regress a local volume $\mathcal{V}_t$ at frame t, expressing a local radiance field. We propose to leverage the mean and variance of the per-voxel features in $\mathcal{U}_i$ computed across neighboring viewpoints; such operations have been widely used in building cost volumes in MVS-based techniques [205, 23], where the mean can fuse per-view appearance information and the variance provides rich correspondence cues for geometry reasoning. These two operation are also invariant to the number/order of input; in our case, this naturally handles voxels that have different numbers of visible viewpoints. We use a deep neural network J to process the mean and variance features per voxel to regress the per-view

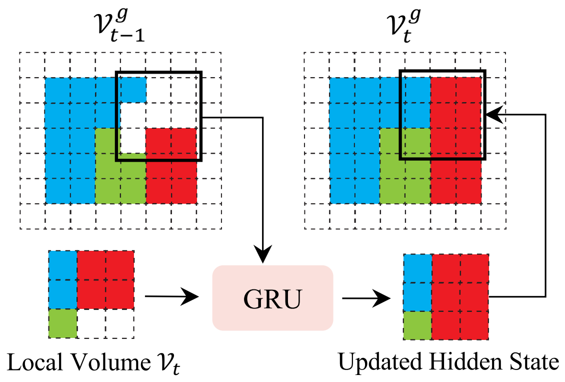

Figure 2.3. A 2D example illustrating the GRU fusion step. The hidden state, which is also the global feature volume $\mathcal{V}_{t-1}^g$, is adaptively updated by aggregating new information in the incoming local feature volume $\mathcal{V}_t$.

reconstruction by

$$\mathcal{V}_t = J([\mathrm{Mean}_{i \in \mathcal{N}_t} \mathcal{U}_i,\ \mathrm{Var}_{i \in \mathcal{N}_t} \mathcal{U}_i]), \tag{2.3}$$

Here $\mathcal{N}_t$ represents all K neighboring viewpoints used at frame t; Mean and Var represent element-wise average and variance operation, respectively.

Essentially, we regress the local radiance field from the features across neighboring views. This is similar to MVSNeRF [23]. However, unlike MVSNeRF that considers only local reconstruction and builds perspective frustrum volumes for small-baseline rendering, we leverage these local volumes for global large-scale reconstruction and rendering. We build volumes directly in canonical space, naturally providing per-frame voxel inputs for our fusion module.

2.3.3 Fusing Volumes for Global Reconstruction

In order to create a consistent, efficient, and extensible scene reconstruction, we propose to use a global neural volume fusion network to incrementally fuse local feature volumes $\{\mathcal{V}_t\}$ per frame into a global volume $\mathcal{V}^g$.

Fusion. At each frame t, we consider its local sparse volume reconstruction $\mathcal{V}_t$ and the global reconstruction $\mathcal{V}^g_{t-1}$ from the previous frame as recurrent input. We leverage GRUs (Gated Recurrent Unit) [33] with sparse 3D CNNs in our fusion module, allowing our network to learn to recurrently fuse the per-frame local reconstruction and output high-quality global radiance fields. This is expressed by

$$z_t = M_z([\mathcal{V}^g_{t-1}, \mathcal{V}_t]), \tag{2.4}$$

$$r_t = M_r([\mathcal{V}^g_{t-1}, \mathcal{V}_t]), \tag{2.5}$$

$$\tilde{\mathcal{V}}^g_t = M_t([r_t * \mathcal{V}^g_{t-1}, \mathcal{V}_t]), \tag{2.6}$$

$$\mathcal{V}^g_t = (1 - z_t) * \mathcal{V}^g_{t-1} + z_t * \tilde{\mathcal{V}}^g_t, \tag{2.7}$$

where $*$ is the element-wise multiplication, z_t and r_t are the update gate and the reset gate, M_z, M_r and M_t all deep neural networks with sparse 3D convolution layers. As in standard GRU, M_z and M_r are designed with sigmoid activation in the end, while M_t uses tanh, allowing for the entire model sequentially updating the global reconstruction $\mathcal{V}^g_t$ (seen as the hidden state in a GRU) for every input frame. In this process, we only apply the networks on the voxels covered by the local volume $\mathcal{V}_t$; all other voxels in the global volume are kept unchanged. A 2D illustration of this GRU fusion process is shown in Figure 2.3.

Intuitively, the update gate z_t and reset gate r_t in the GRU determine how much information from the previous global volume $\mathcal{V}^g_{t-1}$ as well as how much information from the current local volume $\mathcal{V}_t$ should be incorporated into the new global features. In this way, our module can adaptively improve the global scene reconstruction by filling up holes and refining features while keeping the representation consistent. This fusion process is similar to previous 3D reconstruction pipelines [132, 76, 163] that focus on geometry reconstruction; in contrast, we instead reconstruct neural feature volumes to represent neural radiance fields for volume rendering, leading to photo-realistic novel view synthesis.

Voxel pruning. To maximize the memory and rendering efficiency, we adaptively prune the global volume reconstruction $\mathcal{V}_t^g$ for every frame by removing the non-essential voxels that do not have any scene content inside. We naturally leverage the volume density in each voxel regressed by our radiance field (Equation 2.1), which models the scene geometry. In particular, we prune voxels V if:

$$\min_{i=1...k} \exp(-\sigma(v_i)) > \gamma, v_i \in V, \tag{2.8}$$

where $\{v_i\}_{i=1}^k$ are k uniformly sampled points inside the voxel V, $\sigma(v_i)$ is the predicted density at location v_i, and γ is a pruning threshold. This pruning step is performed in both later training phase and inference phase once we get a global feature volume $\mathcal{V}_t^g$. By doing so, we make our global volume sparser, leading to more efficient reconstruction and rendering.

2.3.4 Training and optimization

Once a radiance field (that is represented by a sparse neural volume as described in Sec. 2.3.1) is reconstructed, our final rendering is achieved via differentiable ray marching using the regressed volume density and view-dependent radiance at any sampled ray points, as is done in NeRF and any other radiance field methods [115, 98]. In this work, our full pipeline is trained, completely depending on the rendering supervision with the ground truth images, without any extra geometry supervision.

In particular, we first train our local reconstruction network and the radiance field decoder (R) with a loss

$$\mathcal{L}_{\text{local}} = \|C_t - \hat{C}\|_2^2, \tag{2.9}$$

where $\hat{C}$ is the ground truth pixel color and C_t represents the rendered pixel color using the local volume $\mathcal{V}_t$ reconstructed at frame t. This makes the network learn to predict reasonable local neural volumes, which are already renderable and able to produce realistic images locally; it also initializes the radiance field decoder MLP to a reasonable state, which is later shared across local and global volumes. This pre-training allows the local reconstruction module to provide meaningful volume features for the fusion module to utilize in the end-to-end training, effectively facilitating the fusion task. We then train our full pipeline with the local reconstruction network, fusion network, and the radiance field decoder network all together from end to end, using a rendering loss:

$$\mathcal{L}_{\text{fuse}} = \sum_t \|C_t - \hat{C}\|_2^2 + \|C_t^g - \hat{C}\|_2^2, \tag{2.10}$$

where C_t is the pixel color rendered from the local reconstruction $\mathcal{V}_t$ (as is in Equation 2.9) and C_t^g is the color rendered from the global volume $\mathcal{V}_t^g$ after fusing frame t. Basically, we take every intermediate global and local volume ($\mathcal{V}_t$ and $\mathcal{V}_t^g$) at every frame to render novel view images and supervise them with the ground truth. The fusion module thus reasonably learns to fuse local volumes from an arbitrary number of input frames.

After trained, our full network is able to output a high-quality radiance field from direct network inference and produce realistic rendering results (as shown in Figure 2.1). In addition, the reconstructed radiance field as a sparse neural volume can also be easily optimized (fine-tuned) per scene further to boost the rendering quality.

Fine-tuning. To fine-tune the estimated radiance field, we optimize the per-voxel neural features in the sparse volume reconstruction $\mathcal{V}^g$ and the MLP decoder per scene with the captured images, leading to better rendering results. Since our initial reconstruction is already very good, a short period of optimization with less than 25k iterations can usually lead to very high quality, which takes less than 1 hour. This is substantially less optimization time than NeRF and other pure per-scene optimization methods.

In this per-scene optimization stage, we also do a coarse-to-fine reconstruction, similar to NSVF [98]. Basically, after every 10k optimization iterations, we further prune unnecessary voxels (using Equation 2.8) and also subdivide each voxel into 8 sub voxels.

This prune and subdivision step progressively increases the spatial resolution of the neural volume, further improving our final rendering quality.

2.4 Implementation Details

Training datasets. Our training data consists of both large-scale indoor scenes from ScanNet [36] and small objects from DTU [73] and Google Scanned Objects [143]. We randomly sample 100 scenes from ScanNet for training. For DTU, We adopt the training split from PixelNeRF [208], which includes 88 training scenes. In addition, we use the object-centric synthetic renderings of 1,023 models from the Google Scanned Objects [143] generated by [180]. Our training data includes various camera setups and scene types, enabling our model to generalize to all kinds of scenarios. We demonstrate that our model can effectively work on large-scale indoor scenes, as well as scenes of objects.

Training details. For each input image sequence, we uniformly sample key frames from the full sequences for the input frames in network training. For object-centric datasets, we sample 16 views for each scene, and for ScanNet, we sample $2\% - 5\%$ of the full sequence as key frames. All other frames are used for supervision. We use $K = 3$ neighboring views for each input frame, for local volume reconstruction. For video sequence captured by a monocular camera, such as the scenes in ScanNet, we directly take the 3 neighboring key frames temporally. For other datasets, we select the 3 spatially closest viewpoints, in terms of both viewing location and direction.

Model details. The sparse volumes and networks are implemented with torchsparse [166]. We train our model using Adam optimizer with an initial learning rate of 0.003. We train our network with 2 NVIDIA 2080Ti GPUs for 3 days. During inference, our network processes frames from ScanNet sequences in real-time at 22 FPS. The final model takes 38 seconds on average to render a 640×480 image on ScanNet. We follow [163] to use a modified MnasNet [165] pretrained from ImageNet [149] as the 2D encoder. The output feature channel number from the 2D encoder is 64. The direction encoder G is a 5 layer MLP with 16-channel output. The SparseConvNet J has 5 SparseConv layers implemented with torchsparse [166]. Each of M_z, M_r, M_t has 3 SparseConv layers. All networks use ReLU as activation layers. All features in the feature volumes (global $\mathcal{V}_t^g$ or local $\mathcal{V}_t$) have 16 channels. We apply positional encoding to the volume feature with maximum frequency $L = 5$ before feeding them into the volume renderer. The initial voxel size is set manually for different datasets, specifically $40mm$ for large-scale datasets *e.g.* ScanNet [36], and $4mm$ for object-centric datasets *e.g.* NeRF Synthetic [115] and Google Scanned Objects [143]. When unprojecting 2D features into local feature volumes, we build view frustums with max depth $d_{\max} = 3m$. The pruning threshold γ is set to 0.6 for all experiments. Nearest-neighbor interpolation is used for all coarse-to-fine fine-tuning experiments. The code will be released after the review period.

2.5 Experiments

In this section, we evaluate the our model on various datasets. For all results, we denote our results from direct network inference as Ours and our results after per-scene fine-tuning as Ours_{ft} in all figure and tables. Similar labels are applied to IBRNet and other generalizing methods.

Baselines. We compare our method against the state-of-the-art NeRF methods on novel view synthesis including per-scene optimization methods, such as NeRF [115], NVSF [98], and NerfingMVS [185], and methods that can generalize to new scenes, such as PixelNeRF [208], IBRNet [180], and MVSNeRF [23].

To achieve fair and accurate comparisons, we run our method on the same experiment settings in previous papers, and we try our best to directly use the reported official quantitative results in previous papers or use the official code to run the experiments. We find that the official NeRF and IBRNet code can easily run and work on different

Table 2.1. Quantitative comparisons on the ScanNet dataset [36]. We follow the same experiment settings as in NeRFingMVS [185] and report the error metrics including PSNR (higher is better), SSIM (higher is better) and LPIPS (lower is better). Note that, our diret inference results are better than IBRNet [180]. Our fine-tuning results achieve the best numbers in all three metrics.

Method	Settings	PSNR↑	SSIM↑	LPIPS↓
IBRNet	No per-scene optimization	21.19	0.786	0.358
Ours		**22.99**	**0.838**	**0.335**
NeRF	Per-scene optimization	24.04	0.860	0.334
NSVF		26.01	0.881	-
NeRFingMVS		26.37	0.903	0.245
IBRNet$_{\text{ft-1.5h}}$		25.14	0.871	0.266
Ours$_{\text{ft-1h}}$		**26.49**	**0.915**	**0.209**

datasets, producing corresponding images. We demonstrate visual comparison with these two methods across the three testing sets in Figure 2.4. On the other hand, NSVF is very hard to run without enough GPU memory; their official models are optimized on a V100 GPU that has 32G memory; we found it impractical to generate their corresponding results with our resources. We therefore only include NSVF's quantitative results whenever they are reported previously. Besides, the recent MVSNeRF [23] is a very relevant technique, but it is designed to take a fixed number of three nearby views as its network input; as a result, it cannot support large-baseline rendering or arbitrary number of input images for inference. We therefore only compare with MVSNeRF on the DTU dataset in the same experiment setting used in their paper.

Large-scale scenes in ScanNet. We follow the same training and evaluation scheme as described in NerfingMVS [185] for the comparison on ScanNet. We tested our model on the 8 testing scenes used in their paper. From Table 2.1, we can see that our recurrent neural reconstruction network generates significantly better results than IBRNet via direct network inference. After fine-tuning for only a short period of 1 hour, the quality of our results is further boosted significantly, leading to the best PSNR, SSIM and LIPIPS in all

Table 2.2. Quantitative comparison on the NeRF Synthetic dataset [115]. Our model is able to generate better results than IBRNet in both direct inference and fine-tuning settings. Our model after 1 hour fine-tuning achieves comparable performance to the state-of-the-art per-scene overfitting methods such as NeRF [115] and NVSF [98].

Method	Settings	PSNR↑	SSIM↑	LPIPS↓
IBRNet	No per-scene optimization	**25.51**	0.916	0.100
Ours		**25.47**	**0.922**	**0.093**
NeRF	Per-scene optimization	31.01	0.947	0.081
NSVF		**31.75**	**0.954**	**0.048**
IBRNet$_{\text{ft-1.5h}}$		28.19	0.943	0.072
Ours$_{\text{ft-1h}}$		31.25	**0.953**	0.069

compared methods. Note that, the per-scene optimization methods like NeRF, NeRFing MVS and NSVF require substantially longer per-scene optimization time but are still outperformed by our method. Our approach is impressively better than NSVF in this case though both methods have similar final radiance field representation; this indicates that the data priors learned by our recurrent neural network can effectively help the reconstruction and lead to reasonable initial radiance fields, even benefiting the per-scene fine-tuning process.

As shown in Figure 2.4, our results on these large-scale scenes are of very high visual quality. Our results are visually much better than the IBRNet's results from both direct inference and per-scene fine-tuning. IBRNet generates tearing artifacts since it performs image-based rendering and can only aggregate a small set of local neighboring views due to limited GPU memory. In contrast, our model learns a unified 3D representation in the canonical space with a recurrent module that is able to efficiently aggregate per-view information across all input views, leading to significantly better rendering quality with better across-view consistency. Note that, even our direct inference renderings are already very realistic and contain few noticeable artifacts; they are arguably comparable to the rendering results of NeRF which require long per-scene optimization. Our approach achieves highly efficient and highly accurate large-scale radiance field reconstruction.

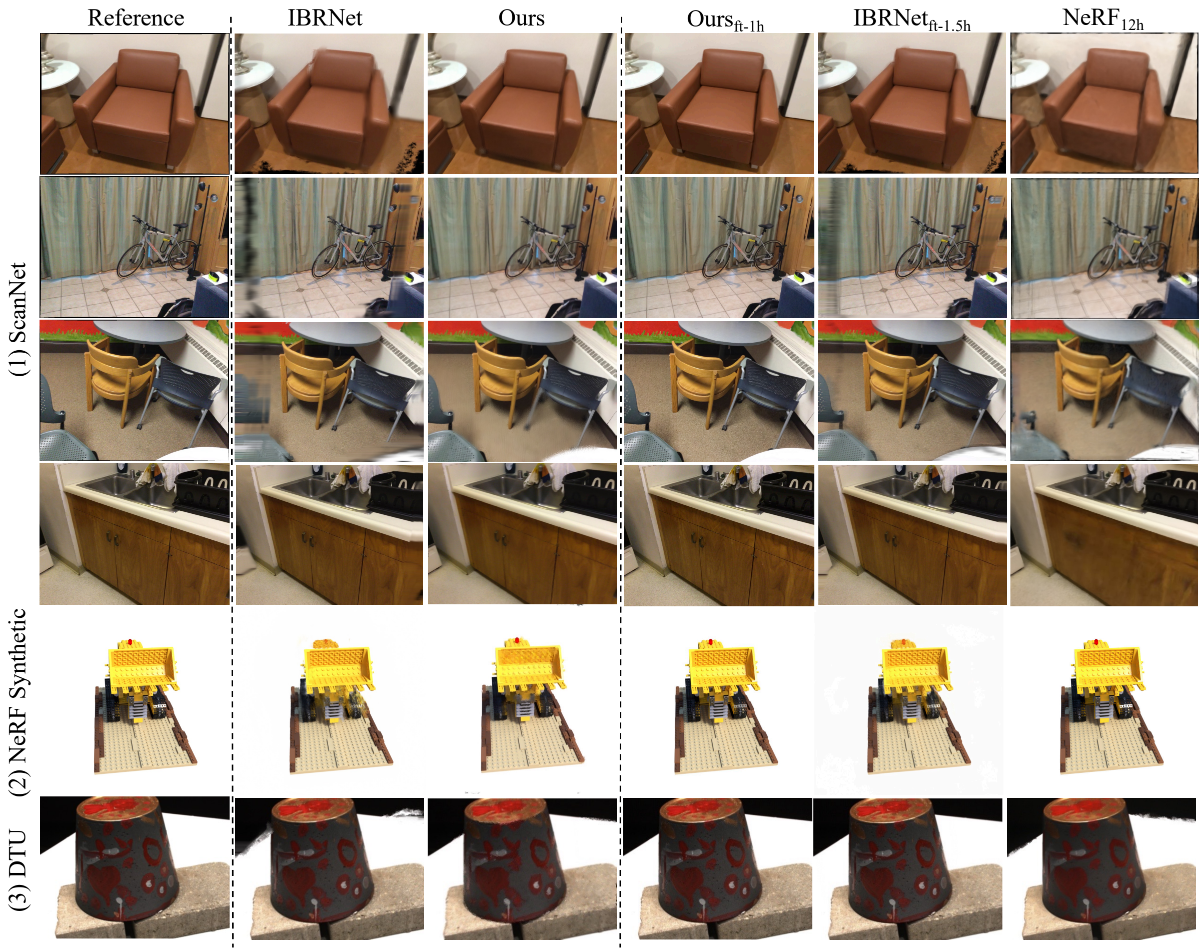

Figure 2.4. Qualitative comparisons of rendering quality on diverse scenes between our method and state-of-the-art method. Our method achieves better performance than state-of-the-art generalizable method IBRNet [180] in both the direct-inference and the fine-tuned settings, where IBRNet generates results with obvious blurry and tearing artifacts. Our model fine-tuned for 1 hour can generate even better results than NeRF [115] that requires 12 hours of training, especially on large-scale scenes from ScanNet [36].

Table 2.3. Quantitative comparisons on the DTU dataset [72]. Our model is able to generate good results under this difficult setting where only 3 input views are given for the direct inference. Our fine-tuning results outperforms other methods in all three metrics.

Method	Settings	PSNR↑	SSIM↑	LPIPS↓
PixelNeRF	No per-scene optimization	19.31	0.789	0.382
IBRNet		26.04	0.917	0.190
MVSNeRF		**26.63**	**0.931**	**0.168**
Ours		26.19	0.922	0.177
NeRF	Per-scene optimization	27.01	0.902	0.263
IBRNet$_{\text{ft-1.5h}}$		31.35	0.956	0.131
MVSNeRF$_{\text{ft-15min}}$		28.50	0.933	0.179
Ours$_{\text{ft-1h}}$		**31.79**	**0.962**	**0.119**

NeRF Synthetic. Our method also works well on small-scale scenes. We conduct experiments on the NeRF Synthetic 360° dataset, and apply the same evaluation setting as in [115]. As shown in Table 2.2, without per-scene fine-tuning, our model generates results that are comparable to IBRNet; however, fine-tuning significantly boosts the performance of our model, leading to high accuracy that is much superior to the fine-tuned IBRNet. In practice, IBRNet suffers from the sparsely distributed input views of the dataset with large baselines, where interpolating neighboring views are not effective to synthesize realistic novel view images. Our fine-tuned model also achieves similar performance when compared to per-scene optimization methods [115, 98], while ours is optimized for only 1 hour, substantially less than the time other methods require.

DTU. To show that our method works with a small number of input views with small baselines. We also evaluate our model on the DTU dataset, following the experiment settings in MVSNeRF [23], where only 3 views are provided for the setting without per-scene optimization and 16 more views are provided in the per-scene optimization setting. Here we also compare with PixelNeRF [208], which is specifically trained for the DTU dataset in their paper. As demonstrated in Table 2.3 and Figure 2.4, similar to previous results, our model generalizes well to the testing scenes and can be efficiently

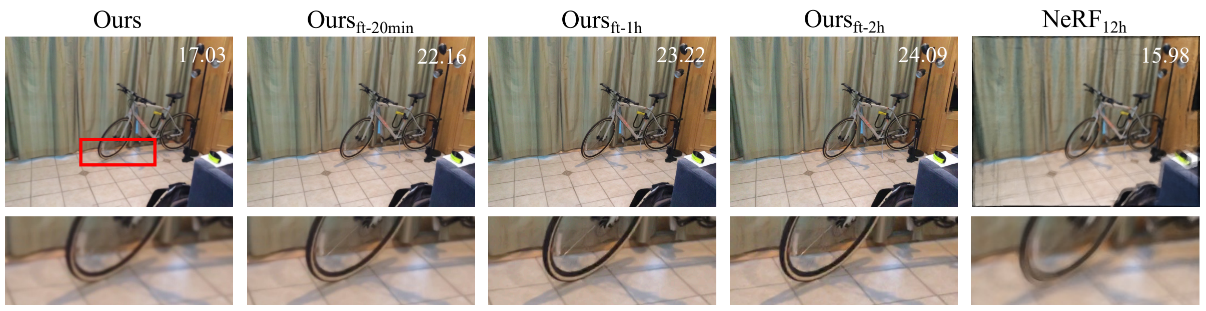

Figure 2.5. Effect of fine-tuning. We show the results with our model with different fine-tuning duration. The first column is our results without fine-tuning. Note that our direct inference result have outperformed NeRF and the results from fine-tuning contain significantly more details. PSNRs are shown at top right of each image.

fine-tuned to outperform NeRF and other generalizable methods including IBRNet [180] and MVSNeRF [23].

2.6 Model Analysis and Discussions

Effect of input view numbers. Figure 2.6 shows the comparison when sample different numbers of neighboring views to build the feature volume. Our learned global feature fusion module is able to effectively fuse information from different views. Note that, when using more input views, our method could produce sharper details in the rendered images, which are very close to the reference.

Effect of fine-tuning. Figure 2.5 shows the visual quality of our method when fine-tuning for different time period. Note that our direct inference result have outperformed NeRF, and the results from fine-tuning contain significantly more details. PSNRs are shown at top right of each image.

Geometry reconstruction. Following [115, 23], we evaluate our geometry reconstruction quality on the DTU dataset [73] by comparing depth reconstruction results generated from the volume density by a weighted sum of the depth values of the sampled points on marched rays. We use 16 input views and compare the depth quality on both input views

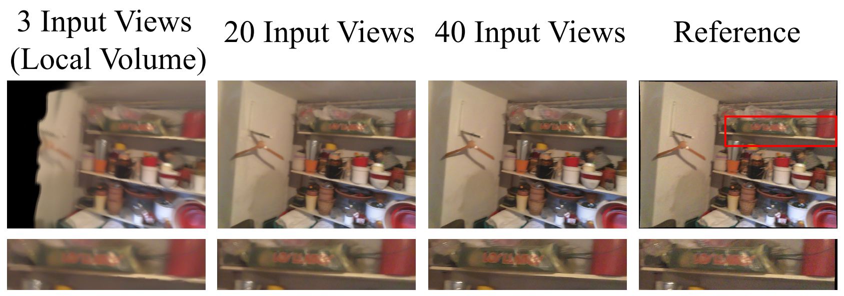

Figure 2.6. Effect of input view numbers. We show the results when sample different numbers of neighboring views to build the feature volume. All results are from the pretrained model without any further fine-tuning. It worth noting that when using more input views, the method could produce sharp details comparable or even better than the reference image.

Table 2.4. Geometry reconstruction. We evaluate depth reconstruction on the DTU testing set and compare with other two neural rendering methods PixelNeRF [208] and IBRNet [180]. Our method significantly outperforms other neural rendering methods and achieves high depth accuracy. The two numbers of each item refers to the depth at *input* / *novel* views.

Method	Abs Err↓	Acc (8mm)↑
PixelNeRF [208]	0.205/0.211	0.096/0.089
IBRNet [180]	1.123/1.324	0.000/0.000
Ours	**0.034/0.036**	**0.722/0.709**

and novel views with 2 common metrics. The results are shown in Table 2.4. Thanks to the explicit geometry modeling in our pipeline, our approach achieves significantly more accurate geometry than other neural rendering methods.

2.7 Limitations

Our approach currently focuses on handling large-scale indoor scenes, but might not be efficient on handling scenes that have foreground objects with distant background, which might appear in unbounded outdoor scenes. This is because we consider a uniform grid for the entire scene, similar to [98]. This can be potentially addressed in the future by doing per-view reconstruction in disparity space or applying spherical coordinates for regions at long distances (similar to [219]). Our method relies on multi-view correspondence; hence, extreme camera poses without enough parallax could lead to problems, which cannot be addressed by any MVS-based techniques. For our current pipeline, we simply sample input frames uniformly, because the camera motion in ScanNet has enough translation. However, a more careful input view selection technique that accounts for relative camera poses may be necessary in practice to address various types of camera motions.

2.8 Conclusion

In this work, we present a novel neural approach that can achieve fast, large-scale, and high-quality scene reconstruction for photo-realistic rendering. In contrast to traditional TSDF-based reconstruction, we reconstruct scenes as volumetric radiance fields, leading to photo-realistic view synthesis results. Our approach leverages a novel recurrent neural network to process the input image sequence and incrementally reconstruct a global large-scale radiance field by reconstructing and fusing per-frame local radiance fields. We demonstrate that our approach can achieve the state-of-the-art rendering quality for large-scale indoor scenes from ScanNet while taking substantially less reconstruction time.

Acknowledgements

Chapter 2, in full, is a reprint of the material published in the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR): “NeRFusion: Fusing Radiance Fields for Large-Scale Scene Reconstruction” (Xiaoshuai Zhang, Sai Bi, Kalyan Sunkavalli, Hao Su, and Zexiang Xu). The dissertation author was the primary investigator and author of this paper.

Chapter 3

Compact and Efficient 3D Scene Representation with Local Radiance Fields

3.1 Introduction

This paper aims to produce a compact, efficient, and comprehensive 3D scene representation from only 2D images. Ideally, the representation should reconstruct appearances, infer semantics, and separate object instances, so that it can be used in a variety of computer vision and robotics tasks, including 2D and 3D panoptic segmentation, interactive scene editing, and novel view synthesis.

Many previous approaches have attempted to generate rich 3D scene representations from images. PanopticFusion [118] produces 3D panoptic labels from images, though it requires input depth measurements from specialized sensors. NeRF [116] and its descendants [117, 8, 9, 142] produce 3D density and radiance fields that are useful for novel view synthesis, surface reconstruction, semantic segmentation [227, 173], and panoptic segmentation [85, 13]. However, existing approaches require 3D ground truth supervision, are inefficient, or do not handle object instances.

We propose *nerflets*, a 3D scene representation with multiple local neural fields that are optimized jointly to describe the appearance, density, semantics, and object instances

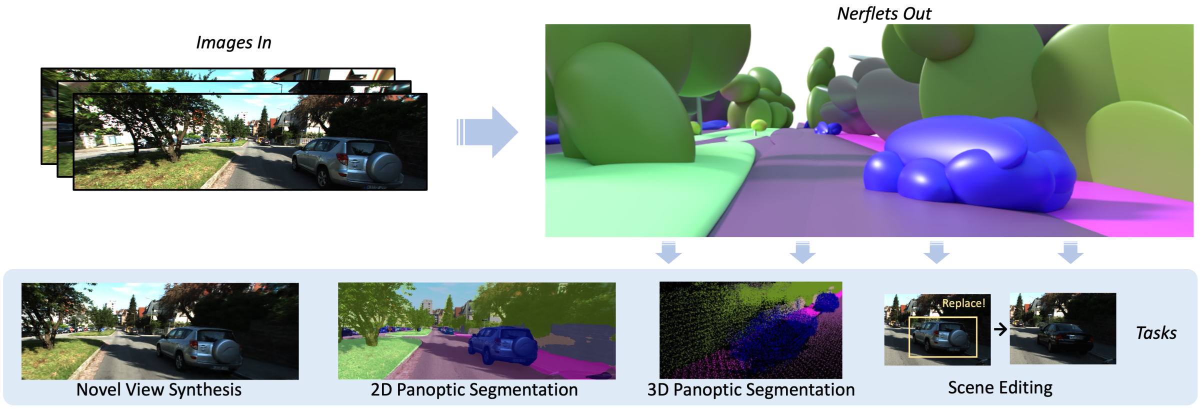

Figure 3.1. We propose to represent the scene with a set of local neural radiance fields, named nerflets, which are trained with only 2D supervision. Our representation is not only useful for 2D tasks such as novel view synthesis and panoptic segmentation, but also capable of solving 3D-oriented tasks such as 3D segmentation and scene editing. The key idea is our learned structured decomposition (top right).

in a scene (Figure 3.1). Nerflets constitute a *structured* and *irregular* representation– each is parameterized by a 3D center, a 3D XYZ rotation, and 3 (per-axis) radii in a 9-DOF coordinate frame. The influence of every nerflet is modulated by a radial basis function (RBF) which falls off with increasing distance from the nerflet center according to its orientation and radii, ensuring that each nerflet contributes to a local part of the scene. Within that region of influence, each nerflet has a miniature MLP to estimate density and radiance. It also stores one semantic logit vector describing the category (e.g., “car”) of the nerflet, and one instance label indicating which real-world object it belongs to (e.g., “the third car”). In Figure 3.1, each ellipsoid is a single nerflet, and they are colored according to their semantics.

A scene can contain any number of nerflets, they may be placed anywhere in space, and they may overlap, which provides the flexibility to model complex, sparse 3D scenes efficiently. Since multiple nerflets can have the same instance label, they can combine to represent the density and radiance distributions of complex object instances. Conversely, since each nerflet has only one instance label, the nerflets provide a complete decomposition of the scene into real-world objects. Nerflets therefore provide a 3D panoptic decomposition of a scene that can be rendered and edited.

Synthesizing images using nerflets proceeds with density-based volume rendering just as in NeRF [116]. However, instead of evaluating one large MLP at each point sample along a ray, we evaluate the small MLPs of only the nerflets near a sample. We average the results, weighting by the influence each nerflet has over that sample. The rendering is fully-differentiable with respect to all continuous parameters of the nerflets. Fitting the nerflet representation is performed from a set of posed RGB images with a single training stage. After training, instance labels are assigned based on the scene structure, completing the representation.

Experiments with indoor and outdoor datasets confirm the main benefits of nerflets. We find that: 1) Sparsity encourages the optimizer to decompose the scene into nerflets with consistent projections into novel panoptic images (Section 3.5.2); 2) Semantic supervision can be beneficial to novel view synthesis (Section 3.6); 3) Structure encourages efficiency, compactness, and scalability (Section 3.3.4); and 4) the explicit decomposition of a scene improves human interpretability for easy interactive editing, including adding and removing objects (Section 3.5.3). These benefits enable state-of-the-art performance on the KITTI360 [94] novel semantic view synthesis benchmark, competitive performance on ScanNet 3D panoptic segmentation tasks with more limited supervision, and an interactive 3D editing tool that leverages the efficiency and 3D decomposition of nerflets.

The following summarizes our main contributions:

- We propose a novel 3D scene representation made of small, posed, local neural fields named *nerflets*.
- The pose, shape, panoptic, and appearance information of nerflets are all fit jointly in a single training stage, resulting in a comprehensive learned 3D decomposition from real RGB images of indoor or outdoor scenes.
- We test nerflets on multiple tasks- novel view synthesis, panoptic view synthesis, 3D

panoptic segmentation and reconstruction, and interactive editing.

- We achieve 1st place on the KITTI-360 semantic novel view synthesis leaderboard.

3.2 Related Work

Recently, the success of deep learning approaches for both computer vision and graphics tasks has enabled researchers to reconstruct and reason about 3D scenes under various settings. We review related work on segmentation and neural field based scene representations.

Semantic, Instance, and Panoptic Segmentation: There are many methods designed for semantic, instance, and/or panoptic [82] segmentation. The most popular approaches are fully-supervised and operate within a single input data modality. For example, 2D approaches [147, 108, 7, 27, 225, 226, 57, 105, 217] are usually based on CNN or transformer backbones and associate each pixel in an image with certain semantic or instance labels. We leverage a trained 2D panoptic model, Panoptic Deeplab [31], in our framework.

Similar frameworks have been proposed to solve 3D segmentation tasks for 3D point clouds [131, 135, 136, 154, 168], meshes [52, 68], voxel grids [49, 159], and octrees [146]. However, these methods typically require a large amount of annotated 3D data, which is expensive to obtain.

To avoid the need for 3D annotations, several multiview fusion approaches have explored aggregating 2D image semantic features onto a pointcloud or mesh using weighted averaging [61, 87, 172, 5], conditional random fields (CRFs) [86, 114], and bayesian fusions [111, 172, 214]. There have also been approaches like 2D3DNet [48] that combine both 2D mutiview fusion with a 3D model.

In contrast to these methods, ours builds a complete 3D representation including geometry, appearance, semantic, and instance information from only 2D inputs and without any input 3D substrate such as a mesh or a point cloud.

Scene Understanding with NeRF: NeRF [116] and subsequent work [85, 125, 202, 228, 173, 83] show the promise of neural radiance fields for tasks beyond novel view synthesis, including 3D reconstruction, semantic segmentation, and panoptic segmentation. For example, SemanticNeRF [228], and NeSF [173] are useful for semantic understanding, but do not consider object instances. DFF [83] leverages the power of large language models for semantic understanding, but similarly does not produce object instances. ObjectNeRF [202] and NSG [125] are useful for object editing, but do not produce a full panoptic decomposition or support efficient interactive editing. None of these methods produce a complete scene representation in the way that nerflets do.

Panoptic Neural Fields (PNF) [85] is very relevant to our paper as it supports both semantic scene understanding and object-level scene editing. PNF first runs a 3D object detector and then a tracker to create an input set of object tracks. They then fit an individual MLP for each object track and another special "stuff" MLP for the remainder of the scene. This is a compelling and effective approach which supports moving objects, but it does not solve our target problem. It 1) requires expensive ground truth 3D supervision for the detector and tracker that are only available for some classes, and 2) has a fixed 3D scene decomposition that is provided by the input tracker results. This last point means that it can fail when the detector or tracker fails, even if a multi-view analysis-by-synthesis appearance loss, like in NeRF, would have been able to force a correct prediction by requiring all pixels to be described by some object instance. By comparison, nerflets require only 2D supervision, support any class for which 2D panoptic segmentations are available, and optimize most parameters jointly, improving efficiency and instance recall.

DM-NeRF [13] is highly related concurrent work. It learns an object decomposition of a scene, but does not provide the explicit structure, full panoptic decomposition, or easy interactive editing of nerflets. In particular, a large MLP decodes spatial positions to object identity vectors, and editing therefore requires careful consideration– an inverse query algorithm [13]. By contrast, nerflets can be edited directly as geometric primitives. We compare quantitatively to both PNF and DM-NeRF in Section 3.5.

Structured NeRF Representations: One of the key advantages of nerflets is their irregular structure, which has been investigated in other contexts. Many existing approaches exploit structure for efficiency [117, 142, 162, 187, 107], compactness [98], scalability [169, 222], human-interpretability [125], parsimony [46], or editability[85, 202]. For example, Kilo-NeRF [142] and DiVER [187] exploit regular grids of MLPs or features to improve the efficiency of NeRF novel view synthesis. MVP [107] builds an irregular primitive-based representation for real-time portrait rendering, but requires explicit scene geometry inputs to initialize the primitives and freezes their location after initialization. We take inspiration from these approaches, which achieve impressive performance through local structure, and apply their insights to panoptic segmentation and editing. Unlike these approaches, nerflets can conform to an object's extent and then move as it is edited. In the future, more benefits of exploring irregular NeRF representations could include tracking a moving object or allowing for a consistent local coordinate frame for learning 3D priors.

3.3 Method

This section introduces our nerflet scene representation and our training and rendering method. As in NeRF [116], the input to our method is a set of posed 2D RGB images. We first run an off-the-shelf 2D panoptic segmentation model [31] to generate predicted 2D semantic and instance images, which we use as a target during the optimization. Next, we optimize our core nerflet representation (Section 3.3.1) to convergence on photometric, semantic, instance, and regularization losses applied to images rendered with volumetric ray-marching [116] (Section 3.3.2). Finally, we assign instance labels to the nerflets based on the learned decomposition (Section 3.3.3), at which point

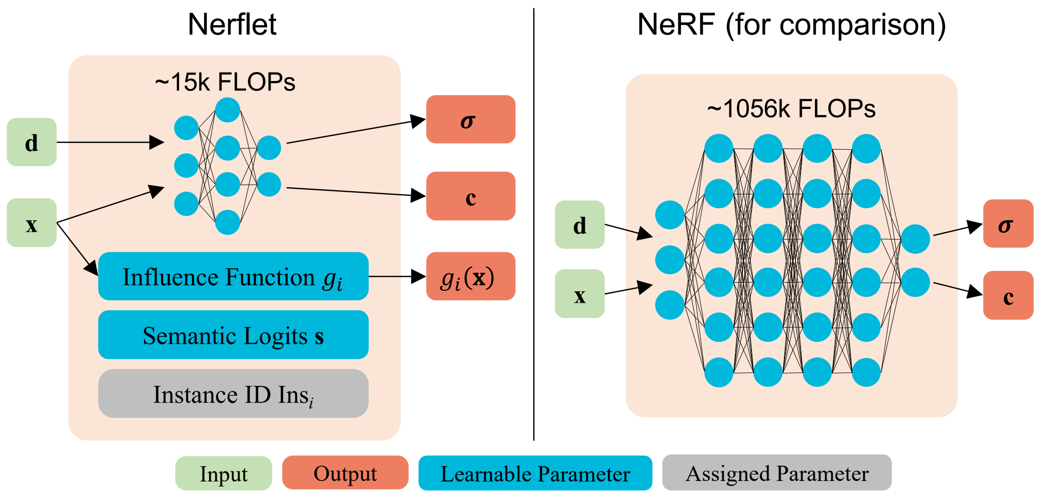

Figure 3.2. Information maintained by a nerflet and NeRF. Compared to NeRF, a nerflet focuses only on a small portion of the scene determined by its influence function g (Equation 3.1), and thus it uses a miniature MLP to fit density σ and color $\mathbf{c}$. Each nerflet also maintains a single semantic logit vector $\mathbf{s}_i$ and an assigned instance ID Ins_i. Together these parameters comprise a compact building block for our scene representation.

the representation is complete and ready for rendering or editing.

3.3.1 Scene Representation

The core novelty of our framework is the nerflet scene representation. Nerflets are a structured representation, where the output radiance and panoptic fields are defined by blending the values produced by N individual nerflets.

Nerflet definition: Each nerflet stores local geometry, appearance, semantic, and instance information. As shown in Figure. 3.2, a nerflet i has 1) position and orientation parameters that define its influence function g_i over space, 2) its own tiny MLP f_i generating both density and radiance values, 3) a single semantic logit vector $\mathbf{s}_i$ storing its semantic information directly, and 4) an associated instance ID Ins_i. Compared to other semantics-aware NeRF methods [227, 173], we use a single logit vector to represent local semantic information instead of training an MLP to encode semantics. This aligns with our goal that a single nerflet should not span multiple classes or instances, and has the additional benefits of reducing the capacity burden of the MLP and providing a natural inductive bias towards 3D spatial consistency.

Pose and influence: Each nerflet has 9 pose parameters– a 3D center $\boldsymbol{\mu}_i$, 3 axis-aligned radii, and 3 rotation angles. We interpret these pose parameters in two ways. First, as a coordinate frame– each nerflet can be rasterized directly for visualization by transforming an ellipsoid into the coordinate frame defined by the nerflet. This is useful for editing and understanding the scene structure (e.g., Figure 3.1). The second way, more critical for rendering, is via an influence function $g_i(\mathbf{x})$ defined by the same 9 pose parameters. g_i is an analytic radial basis function (RBF) based on scaled anisotropic multivariate Gaussians [47, 46]

$$g_i(\mathbf{x}) = \eta \exp\left(-\frac{1}{2\tau}(\mathbf{x}-\boldsymbol{\mu}_i)^T \Sigma_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)\right). \tag{3.1}$$

$\boldsymbol{\mu}_i$ is the center of the basis function and Σ_i is a 6-DOF covariance matrix. The covariance matrix is determined by 3 Euler-angle rotation angles and 3 axis-aligned radii that are the reciprocal of variance along each principal axis. These 9 parameters provide a fast and compact way to evaluate a region of influence for each nerflet without evaluating any neural network. This property is crucial for our fast training and evaluation, which will be introduced later. η is a scaling hyper-parameter set to 5 for all experiments, and τ is a scheduled temperature hyper-parameter used to control the falloff hardness. τ is reduced after each training epoch to minimize overlap between nerflets gradually.

Rendering and blending: Given a scene represented by N nerflets, we can render 2D images with volume rendering, as in NeRF:

$$\hat{C}(\mathbf{r}) = \sum_{k=1}^{K} T_k \alpha_k \mathbf{c}_k, \tag{3.2}$$

$$\text{where} \quad \alpha_k = 1 - \exp(-\sigma_k \delta_k), \tag{3.3}$$

$$T_k = \prod_{j=1}^{k-1}(1-\alpha_j). \tag{3.4}$$

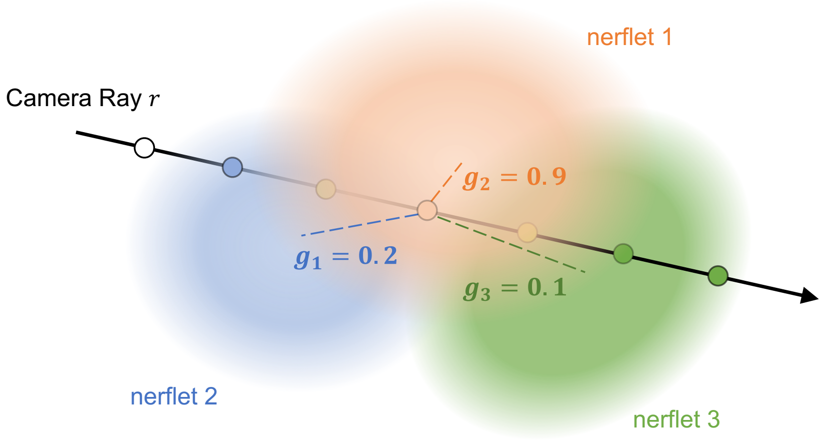

Figure 3.3. Sample and blend method illustration. Results from individual nerflets are mixed based on the influence values g_i determined by the distance-based weighting function. The mixing is smooth but most locations in space are dominated by a single nerflet, even when there is some overlap.

$\hat{C}(\mathbf{r})$ is the final color of ray $\mathbf{r}$, T_k is transmission at the k-th sample along the ray, α_k is the opacity of the sample, $\mathbf{c}_k$ is the color at the sample, δ_k is the thickness of the current sample on the ray, and σ_k is the density at the sample. We denote k for the index of the sample along a ray, K total number of samples and reserve i for the index of the nerflet.

The biggest difference from NeRF is that instead of employing a single large MLP to produce $\mathbf{c}_k$ and σ_k, we combine values produced by individual nerflets using their influence weights (Figure 3.3). We query individual nerflet MLPs f_i at the k-th input sample $(\text{pos}, \text{dir}) = (\mathbf{x}_k, \mathbf{d})$ along the ray, producing producing N values $\sigma_{k,i}$ and $\mathbf{c}_{k,i}$ for nerflets labeled $i \in [1, N]$. We then map the individual nerflet $\sigma_{k,i}$ values to $\alpha_{k,i}$ for rendering using δ_k and Equation 3.2. Finally, we take a weighted average of the N individual nerflet color and α values to produce $\mathbf{c}_k$ and α_k:

$$\mathbf{c}_k = \sum_{i=1}^{N} \hat{g}_i(\mathbf{x}_k)\mathbf{c}_{k,i}, \tag{3.5}$$

$$\alpha_k = \sum_{i=1}^{N} \hat{g}_i(\mathbf{x}_k)\alpha_{k,i}, \tag{3.6}$$

$$\text{where} \quad \hat{g}_i(\mathbf{x}_k) = \frac{g_i(\mathbf{x}_k)}{\sum_{j=1}^{N} g_j(\mathbf{x}_k) + \epsilon}. \tag{3.7}$$

ϵ is a factor allowing smooth decay to zero in empty space, with $\Sigma g_i(\mathbf{x}_k) \sim \epsilon$. After blending, the α_k and $\mathbf{c}_k$ values are used directly for ray marching as in Equation 3.2-3.4 to generate final pixel color values $\hat{C}(\mathbf{r})$, as in NeRF.

While in principle we should evaluate all nerflet MLPs in the scene in this step, as gaussian RBFs have infinite support, we do not do this. Typically, g_i is dominated by one or at most a handful of nerflets that are close to the sample. Therefore, we evaluate only the nearby MLPs, improving performance and scalability. This is implemented with our custom CUDA kernel, which will be introduced in Sec. 3.3.4. More distant nerflets are omitted from the average.

To generate semantic images, we average the per-nerflet semantic logit vector for each point sample in the same way described for color values $\mathbf{c}_{k,i}$ above.

To handle instances, we first compute a *nerflet influence activation* function $\mathbf{w} \in \mathbb{R}^n$ for each point sample:

$$\mathbf{w}(\mathbf{x}) = \text{SoftMax}([\sigma_1 g_1(\mathbf{x}), \ldots, \sigma_N g_N(\mathbf{x})]). \tag{3.8}$$

σ_i is the density evaluation for $\mathbf{x}$ on the i-th nerflet. This value intuitively represents how much influence each nerflet has over a given point sample, and can be accumulated by ray marching to generate a nerflet influence map $W(\mathbf{r})$ for each ray $\mathbf{r}$. $W(\mathbf{r})$ continuously captures which nerflet is dominant for each final pixel value, and is used by our influence loss described in Section 3.3.2. To assign a discrete instance label to a query position or ray, we take $\text{argmax}_{i\in[N]} w(\mathbf{r})$ or $\text{argmax}_{i\in[N]} W(r)$, respectively to get i, then output Ins_i, the instance of that nerflet.

Unbounded scenes: Nerflets support both indoor (bounded) and outdoor (unbounded) scenes. To handle unbounded scenes, we add a single MLP f_{far} to evaluate samples outside a large scene bounding box. We draw M additional samples for these points at the end of the ray, which we concatenate after our blended RGBα values and composite. We use the scheme proposed by Zhang *et al.* [219], with an added semantics branch, though as the content is very distant and nearly directional, many other approaches would likely work well (e.g., an environment map).

3.3.2 Loss Function

During training, we jointly optimize all network parameters as well as the pose of the nerflets. In this way, each nerflet can be pushed by gradients to "move" across the scene, and focus on a specific portion of it. We expect a final decomposition mirroring the scene, with more nerflets on complex objects, and use multiple losses to that end.

The loss function is broken up into rgb, semantic, instance, and regularization terms:

$$\mathcal{L} = \mathcal{L}_{\text{rgb}} + \mathcal{L}_{\text{sem}} + \mathcal{L}_{\text{ins}} + \mathcal{L}_{\text{reg}}. \tag{3.9}$$

$\mathcal{L}_{\textbf{rgb}}$**:** The RGB loss $\mathcal{L}_{\text{rgb}}$ is the mean squared error between the synthesized color $\hat{C}$ and the ground truth color C averaged over a batch of sampled rays, as in the original NeRF. The one change is that we weight this loss with a schedule parameter that is 0.0 at step 0. We gradually increase this value to 1.0 during training to prevent early overfitting to high frequency appearance information.

$\mathcal{L}_{\textbf{sem}}$**:** Our semantic loss $\mathcal{L}_{\text{sem}}$ compares the volume-rendered semantic logits pixel with the Panoptic Deeplab prediction [31]. We use a per-pixel softmax-cross-entropy function for this loss.

$\mathcal{L}_{\mathbf{ins}}$: The instance loss is defined as:

$$\mathcal{L}_{\text{ins}} = -\frac{1}{P}\sum_{(\mathbf{r}_1,\mathbf{r}_2)} ||W(\mathbf{r}_1) - W(\mathbf{r}_2)||_1. \tag{3.10}$$

That is, we sample P ray pairs $(\mathbf{r}_1, \mathbf{r}_2)$ that are from the same class but different instances according to the instance segmentation model prediction, and enforce them to have different influence maps. While this approach is somewhat indirect, well separated nerflets can be easily assigned instance labels (Section 3.3.3), and it has the advantage of avoiding topology issues due to a variable number of instances in the scene while still achieving an analysis-by-synthesis loss targeting the instance decomposition. It is also compatible with the inconsistent instance ID labelings across different 2D panoptic image predictions. Ray pairs $(\mathbf{r}_1, \mathbf{r}_2)$ are chosen within in an $L \times L$ pixel window per-batch for training efficiency.

$\mathcal{L}_{\mathbf{reg}}$: Our regularization loss has several terms to make the structure of the nerflets better mirror the structure of the scene–

$$\mathcal{L}_{\text{reg}} = \mathcal{L}_{\text{density}} + \mathcal{L}_{\text{radii}} + \mathcal{L}_{\ell_1} + \mathcal{L}_{\text{box}}. \tag{3.11}$$

In addition to the intuitions described below, each of these is validated in a knock-out ablation study (Sec. 3.6) and tested on multiple datasets, to reduce the risk of overfitting to one setting.

First, to minimize unnecessary nerflet overlap within objects and reduce scene clutter, we penalize the L_2 norm of the radii of the nerflets ($\mathcal{L}_{radii}$). To encourage sparsity where possible, we penalize the L_1 norm of the nerflet influence values at the sample locations ($\mathcal{L}_{\ell_1}$). We also require nerflets to stay within their scene bounding box, penalizing:

$$\sum_{d \in \{x,y,z\}} \max(x_d - \text{box}_{\max}, \text{box}_{\min} - x_d, 0). \tag{3.12}$$

This reduces risk of "dead" nerflets, where a nerflet is far from the scene content, so it does not contribute to the loss, and therefore would receive no gradient.

Finally, we incorporate a "density" regularization loss $\mathcal{L}_{\text{density}}$, which substantially improves the decomposition quality:

$$\mathcal{L}_{\text{density}} = -\frac{1}{D} \sum_{i=1}^{n} \textstyle\sum_{\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}_i, \Sigma_i)} \sigma(\mathbf{x}). \tag{3.13}$$

$\mathcal{N}(\boldsymbol{\mu}_i, \Sigma_i)$ represents the underlying multivariate gaussian distribution for the i-th nerflet and D is the number of samples drawn. This term rewards a nerflet for creating density near its center location. As a result, nerflets end up centered inside the objects they reconstruct.

3.3.3 Instance Label Assignment

Given an optimized scene representation, we use a greedy merge algorithm to group the nerflets and associate them with actual object instances. We first pick an arbitrary 2D instance image, and render the associated nerflet influence map $W(\mathbf{r})$ for the image. We then assign the nerflets most responsible for rendering each 2D instance to a 3D instance based on it. We proceed to the next image, assigning nerflets to new or existing 3D instances as needed. Because the nerflets have been optimized to project to only a single 2D instance in the training images, this stage is not prone to failure unless the 2D panoptic images disagree strongly.

To assign instance labels for each nerflet, we render the nerflet influence map W_i for each view and compare with corresponding 2D semantic and instance segmentation maps to match each l_i^j (2D object instance or stuff with local id j in view i) to a set of nerflets $M(l_i^j)$. Here $M(\cdot)$ maps an instance ID to its set of associated nerflets. We then create a set of 3D instances $G = \{g_k\}$ according to the segmentation result of the first view – we create a 3D global instance for each detected 2D object instance and also each disjoint stuff labels in the semantic maps. For each new view i, we match l_i^j to the 3D instance g_k if $|M(l_i^j) \cap M(g_k)|/|M(l_i^j)| \geq \delta$, and then update $M(g_k)$ to $M(g_k) \cup M(l_i^j)$. If no match is found in $\{g_k\}$, we create a new 3D instance and insert it into G. Before inserting any new global instance g, we remove all nerflets that are already covered by the global set G from g. By using this first-come-first-serve greedy strategy we always guarantee no nerflet is associated with 2 different global instances. After this step, each nerflet is associated with a global instance ID, and our representation can be used to reason at an instance level effectively.

3.3.4 Efficient Nerflet Evaluation

Top-k Evaluation: Instead of evaluating all nerflet MLPs in a scene as in Equation 3.5-3.8, we use the g_i influence values to filter out distant and irrelevant nerflets in two ways. First, we truncate all nerflet influences below some trivial threshold to 0– there is no need to evaluate the MLP at all if it has limited support. In free space, often all MLPs can be ignored this way. Next, we implement a "top-k" MLP evaluation CUDA kernel that is compatible with the autodiff training and inference framework. This kernel evaluates only the highest influence nerflet MLPs associated with each sample. We use $k = 16$ for both training and visualizations in this work, although even more aggressive pruning is quite similar in image quality (e.g., a difference of only ~ 0.05 PSNR between top-16 and top-3) and provides a substantial reduction in compute. A top-k ablation study is available in the supplemental.

Interactive Visualization and Scene Editing: We develop an interactive visualizer combining CUDA and OpenGL for nerflets that takes advantage of their structure, efficiency, rendering quality, and panoptic decomposition of the scene. Details are available in the supplemental. The key insight enabling efficient evaluation is that nerflets have a good sparsity pattern for acceleration – they are sparse with consistent local structure. We greatly reduce computation with the following two pass approach. In the first pass, we determine where in the volumetric sample grid nerflets have high enough influence to contribute to the final image. In the second pass, we evaluate small spatially adjacent subgrids for a particular nerflet MLP, which is generally high-influence for all samples in the subgrid due to the low spatial frequency of the RBF function. This amortizes the memory bandwidth of loading the MLP layers into shared memory. This approach is not as fast as InstantNGP [117], but is still interactive and has the advantage of mirroring the scene structure.

3.4 Implementation Details

Model Architecture: For all nerflets MLPs f_i, we follow the NeRF architecture [116] but reduce the number of hidden layers from 8 to 4, and reduce the number of hidden dimensions from 256 to 32. We also removed the shortcut connection in the original network. All other architecture details are as in [116]. The background neural field uses NeRF++ [219] style encoding, and its MLP f_{far} is with 6 hidden layers and 128 hidden dimensions. One distinction is that we do perform coarse-to-fine sampling as in [116], but both coarse and fine samples are drawn from a single MLP, not two distinct ones.

Hyper-parameters: We use $N = 512$ nerflets for all experiments in the main paper. The scaling parameter η is set to 5 for all experiments. We initialize the nerflets temperature parameter τ to 1 and multiply τ by 0.9 across the epochs. The smooth decay factor ϵ is set to 10^{-7} for all experiments. We draw 64 samples for coarse level and 128 samples for fine level within the bounding box. For unbounded scenes, we draw 16 coarse samples and 16 fine samples from the background MLP. We increase the weight for $\mathcal{L}_{\text{rgb}}$ from 0.0

to a maximum of 1.0 by the step of 0.2 across epochs to prevent early overfitting to high frequency information. Contrastive ray pairs are sampled within an 32×32 pixel window. The weight for regularization loss $\mathcal{L}_{\text{reg}}$ is set to 0.1. All other losses are with weight 1.0.

Dataset Details: For training on each ScanNet scene, we uniformly sample 20% of the RGB frames for training and 10% of the RGB frames for evaluation– about 200 frames for training and 100 frames for evaluation. For both ScanNet and KITTI-360 scenes, we estimate the scene bounding box with camera extrinsics and normalize the coordinate inputs to $[-0.5, 0.5]$ for all experiments.

One important note about the ScanNet [36] experiments is that 2D ScanNet supervision indirectly comes from 3D. That is because the 2D ScanNet dataset was made by rendering the labeled mesh into images. We do not use this 2D ground truth directly, but PSPNet [225] is trained on it. Here, this is primarily a limitation of the evaluation rather than the method– there are many 2D models that can predict reasonable semantics and instances on ScanNet images, but we want to be able to evaluate against the exact classes present in the 3D ground truth. This does not affect the comparison to other 2D supervised methods, as all receive their supervision from the same 2D model. By comparison KITTI-360 results are purely 2D only, but all quantitative evaluations must be done in image-space.

For KITTI-360 experiments, we use the novel view synthesis split, and compare to Panoptic Neural Fields (PNF) [85], a recent state of the art method for panoptic novel view synthesis (1st on the KITTI-360 leaderboard). To generate 2D panoptic predictions for outdoor scenes, we use a Panoptic DeepLab [31] model trained on COCO [96].

For ScanNet experiments, we evaluate on 8 scenes as in DM-NeRF [13] and compare to recent baselines– DM-NeRF [13], which synthesizes both semantics and instance information, and Semantic-NeRF [227] which synthesizes semantics only. To generate panoptic images for indoor scenes, we use PSPNet [225] and Mask R-CNN [57]. Please see the supplemental for important subtleties regarding how 2D supervision on

ScanNet is achieved.

Interactive Visualizer Details: In order to demonstrate the efficiency of our representation, we have implemented a real-time interactive visualizer for nerflets. Our interactive visualizer allows real-time previewing of nerflet editing results while adjusting the bounding boxes of objects in the scene. The visualizer draws the following components. First, a volume-rendered RGB or depth image at an interactive resolution of up to 320x240. This enables viewing the changes being made to the scene in real time. Second, the nerflets directly, by rendering an ellipsoid per nerflet at a configurable influence threshold. This enables seeing the scene decomposition produced by the nerflets. Third, a dynamic isosurface mesh extracted via marching cubes that updates as the scene is edited, giving some sense of where the nerflets are in relation to the content of the scene. Fourth, a set of boxing box manipulators, one per object instance, with draggable translation and rotation handles. These boxes are instantiated by taking the bounding box of the ellipsoid outline meshes for all nerflets associated with a single instance ID. A transformation matrix that varies per instance is stored and pushed to the nerflets on each edit.

Most of the editor is implemented in OpenGL, with the volume rendering implemented as a sequence of CUDA kernels that execute asynchronously and are transferred to the preview window when ready. In the main paper we report performance numbers for top-1 evaluation, which is often the right compromise for maximizing perceived quality in a given budget (e.g., pixel count can be more important), though interactive framerates with top-16 or top-3 evaluation are possible at somewhat lower resolutions.

3.5 Experiments

In this section, we evaluate our method using 512 nerflets on multiple tasks with two challenging real-world datasets.

Table 3.1. Results on novel view color and semantic synthesis tasks on KITTI-360 [94]. Nerflets achieve similar color synthesis quality and better semantic synthesis quality compared to PNF [85] without any 3D supervision. We also have the best efficiency in terms of worst case kFLOPs.

Method	Appearance PSNR	Semantics mIOU	Worst-Case kFLOPs
PBNR [84] + PSPNet [225]	19.91	65.07	-
FVS [145] + PSPNet [225]	20.00	67.08	-
NeRF [116] + PSPNet [225]	21.18	53.01	~ 1056
Mip-NeRF [8] + PSPNet [225]	21.54	51.15	~ 1056
PNF [85]	**21.91**	<u>74.28</u>	~ 1256
Ours	<u>21.69</u>	**75.07**	~ 244

Table 3.2. ScanNet novel view synthesis quantitative results. Rendered color images and segmentation maps from nerflets have the best quality among all evaluated methods.

Method	PSNR	Semantics mIoU	Instance mAP0.5
PSPNet [225] on GT Image	-	68.43	-
Mask R-CNN [57] on GT Image	-	-	23.53
PSPNet [225] on NeRF Im.	-	46.21	-
Mask R-CNN [57] on NeRF Im.	-	-	14.32
Semantic-NeRF [227]	28.43	71.34	-
DM-NeRF [13]	28.21	70.71	25.12
Ours	**29.12**	**73.63**	**31.32**

Table 3.3. Evaluation of nerflet panoptic performance on ScanNet 3D point cloud labeling task. Nerflets beat some less recent fully 3D-supervised methods with less supervision at both semantic and instance tasks, while also beating the similarly supervised multiview fusion approach.

Supervision	Method	Semantics mIoU	Instance mAP0.5
Pointcloud	MinkowskiNet [34]	**71.92**	-
	3DMV [37]	49.22	-
	PointNet++ [136]	44.54	-
	Mask3D [151]	-	**75.34**
	3D-BoNet [203]	-	46.23
Images	Multiview Fusion [48]	55.23	-
	Ours	**63.94**	48.67

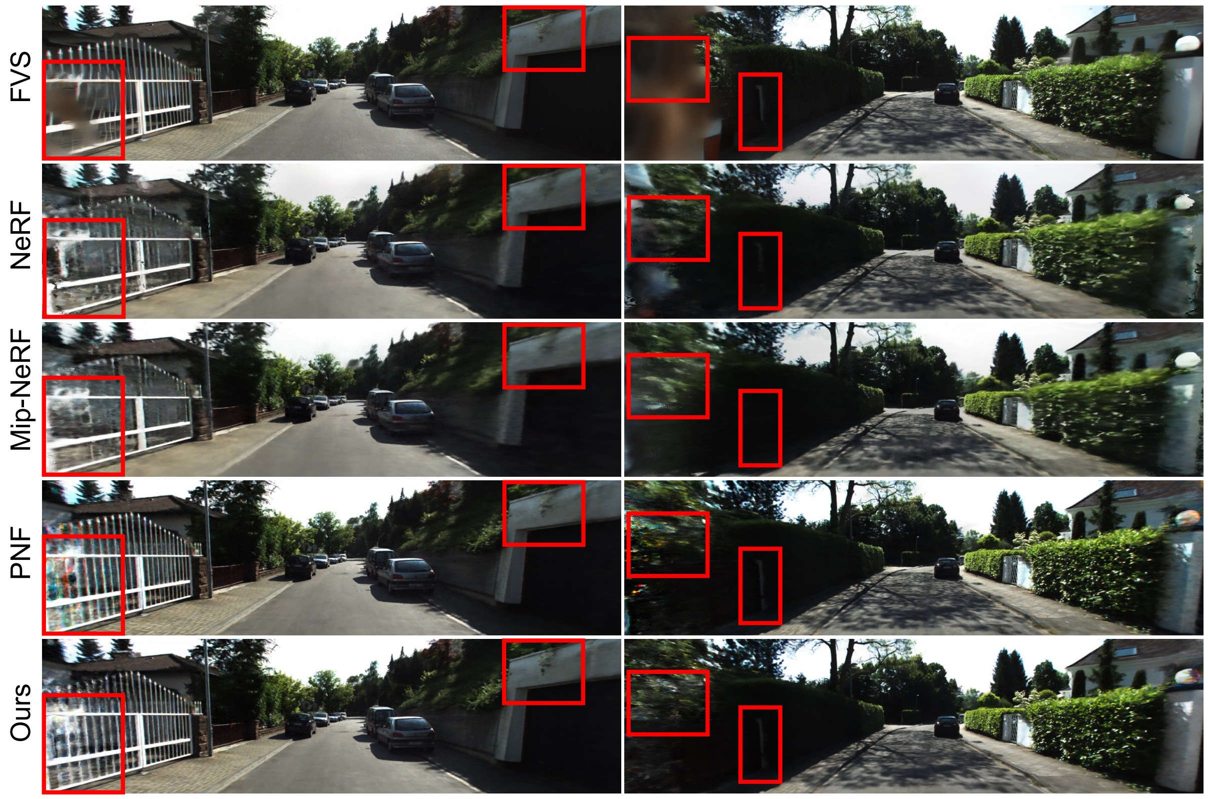

Figure 3.4. Novel view synthesis qualitative comparison. Nerflets outperform NeRF, Mip-NeRF, and FVS, and perform comparably to PNF with better performance in difficult areas (far left), possibly due to explicit spatial allocation of parameters.

3.5.1 Novel View Synthesis

We evaluate the performance of nerflets for novel view image synthesis on both KITTI-360 and ScanNet. As shown in Table 3.1, on KITTI-360, our method achieves better PSNR than all other 2D supervised methods on the leaderboard and is competitive with PNF [85], which utilizes 3D supervision. As shown in Figure 3.4 our visual quality is approximately on par with PNF, and does particularly well in challenging areas. For complex indoor scenes in ScanNet (Table 3.2), we achieve the best performance for novel view synthesis under all settings (with or without instance supervision), including when compared to DM-NeRF [13]. In particular, nerflets achieve better object details (Figure 3.6), likely due to their explicit allocation of parameters to individual object instances.

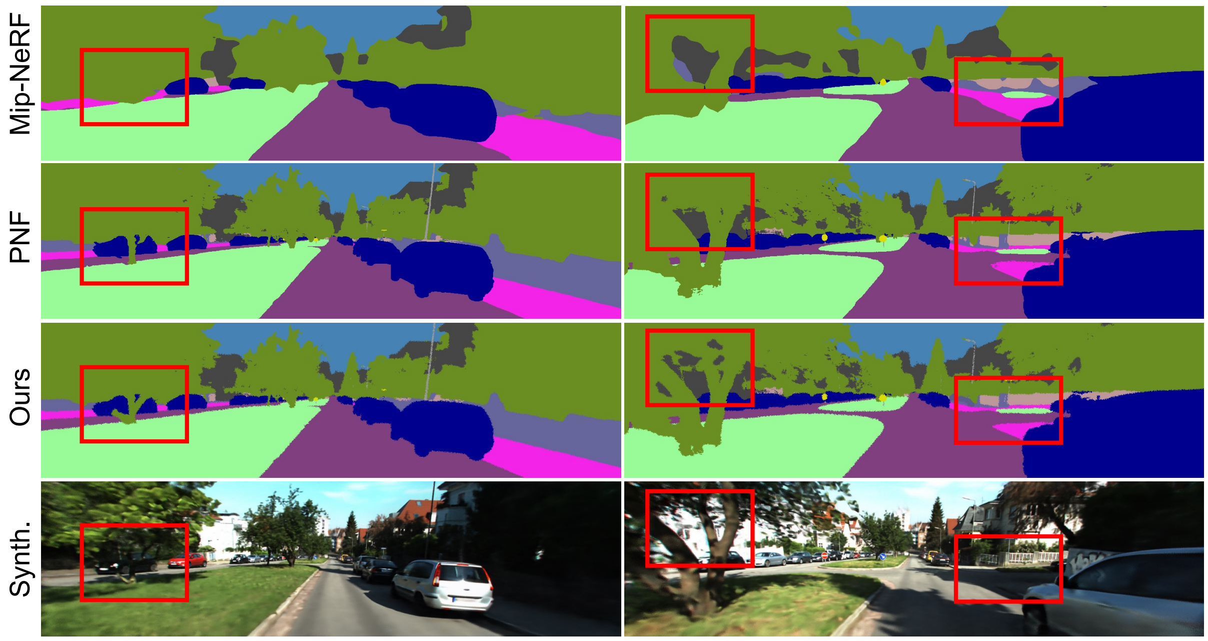

Figure 3.5. Novel view semantic synthesis qualitative comparison. Nerflets outperform other methods, particularly with respect to details and thin structures.

3.5.2 2D Panoptic Segmentation

Nerflets can render semantic and instance segmentations at novel views. We evaluate our 2D panoptic rendering performance quantitatively on both KITTI-360 (Table 3.1) and ScanNet (Table 3.2). On both datasets, nerflets outperform all baselines in terms of semantic mIoU, even compared to the 3D-supervised PNF [85]. On the ScanNet dataset, we also show that nerflets outperform PSPNet [225] and Mask R-CNN [57] in terms of both mIoU and instance mAP, although those methods were used to generate the 2D supervision for our method. This is an indication that nerflets are not only expressive enough to represent the input masks despite their much lower-dimensional semantic parameterization, but also that nerflets are effectively fusing 2D information from multiple views into a better more consistent 3D whole. We further explore this in the supplemental material. Qualitatively, nerflets achieve better, more detailed segmentations compared to baseline methods (Figure 3.5, Figure 3.6), particularly for thin structures.

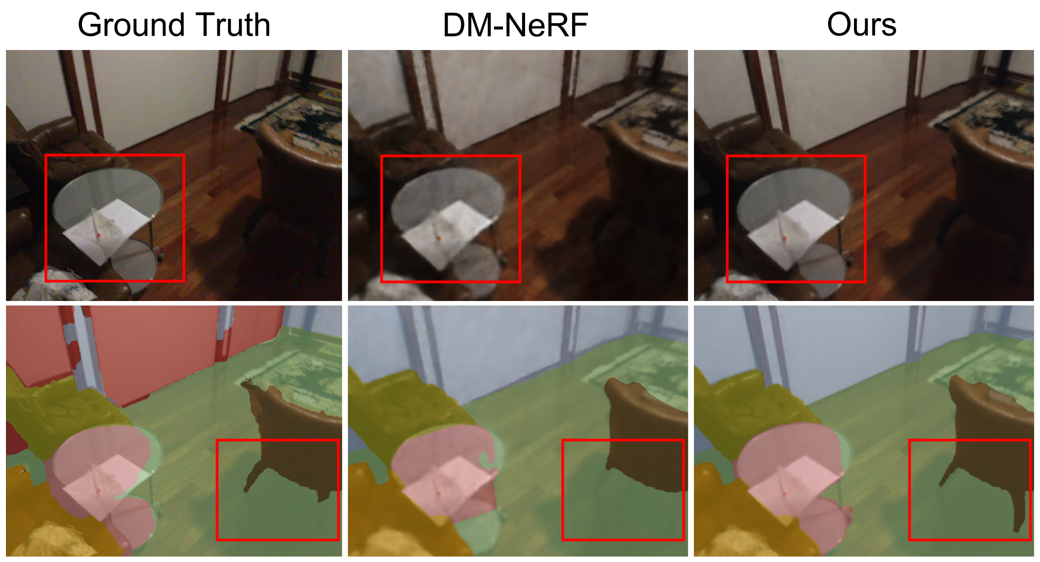

Figure 3.6. ScanNet qualitative result and comparison to DM-NeRF [13]. The comparison is on a ScanNet view synthesis example taken from the DM-NeRF paper. Our results improve in terms of both image and segmentation quality- notice the better image rendering for the glass table, and the better segmentation of the chair legs, which even exceeds the ground truth quality.

3.5.3 Scene Editing

In Figure 3.7 and Figure 3.8, we use the instance labels on nerflets to select individual objects, and then manipulate the nerflet structure directly to edit scenes. No additional optimization is required, and editing can be done while rendering at interactive framerates (please see the video for a demonstration, the results here were rendered by the standard autodiff inference code for the paper). When compared to Object-NeRF on ScanNet (Figure 3.8), nerflets generate cleaner results with more detail, thanks to their explicit structure and alignment with object boundaries. Using nerflets, empty scene regions will not carry any density after deletion, as there is nothing there to evaluate. In Figure 3.7, we demonstrate additional edits on KITTI-360, with similar results. These visualizations help to confirm that indeed, nerflets learn a precise and useful 3D decomposition of the scene.

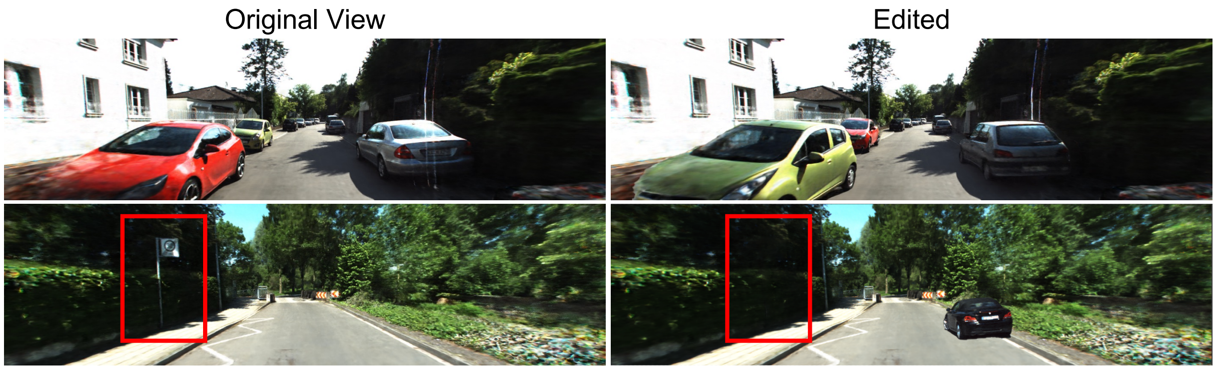

Figure 3.7. KITTI-360 scene editing. We replace cars (top) or removing a sign instance (bottom).

3.5.4 3D Panoptic Reconstruction

In Figure 3.9, we demonstrate the 3D capabilities of nerflets by extracting a panoptically labeled 3D mesh and comparing it to the ground truth. We create point samples on a grid and evaluate their density, semantic and instance information from nerflets. We then estimate point normals using the 5 nearest neighbors and create a mesh with screened Poisson surface reconstruction [77]. The mesh triangles are colored according to the semantic and instance labels of their vertices. We observe that the resulting mesh has both good reconstruction and panoptic quality compared to the ground truth. For example, nerflets even reveal a chair instance that is entirely absent in the ground truth mesh. We demonstrate this quantitatively by transferring nerflet representations to a set of ground truth 3D ScanNet meshes, comparing to existing 3D-labeling approaches in Table 3.3. We observe that nerflets outperform the similarly supervised multi-view fusion baseline, while adding instance capabilities. State of the art directly-3D supervised baselines are still more effective than nerflets when input geometry and a large 3D training corpus are available, but even so nerflets outperform some older 3D semantic and instance segmentation methods.

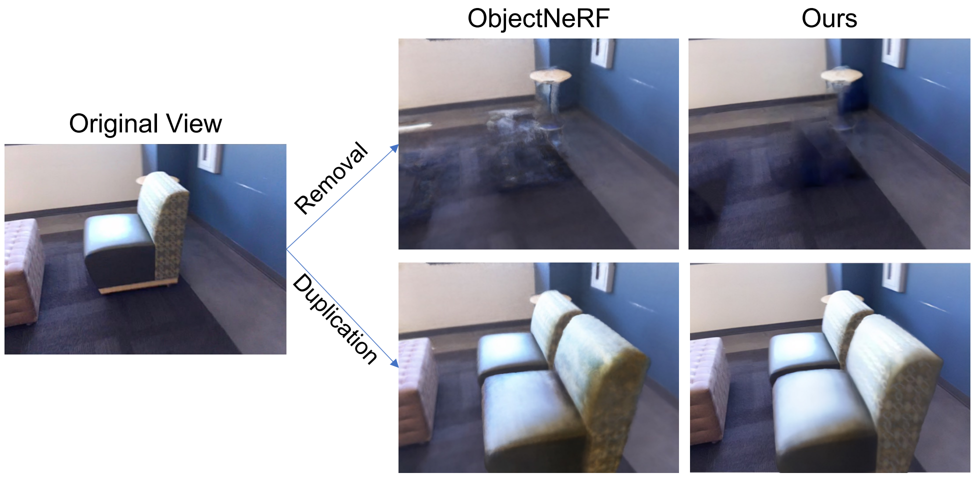

Figure 3.8. Scene editing comparison with ObjectNeRF on a pair of ScanNet images shown in the ObjectNeRF paper. Notice the improved handling of free space during removal and the more accurate texture during duplication, both attributable to the simple "copy-and-paste" nature of nerflets manipulation.

3.6 Model Analysis and Discussions

Scene Decomposition Quality: Our insight was to create an irregular representation that mirrors the structure of the scene. Do nerflets succeed at achieving this scene decomposition? In Figure 3.10, we show RGB and panoptic images alongside the underlying nerflet decomposition that generated them. We visualize nerflets according to its influence function. We draw ellipsoids at influence value $e^{-\frac{1}{2}} \approx 0.607$. We see that indeed, the nerflets do not cross object boundaries, do join together to represent large or complex objects, and do cover the scene content.

Semantics Help Appearance: One key insight about our approach is that the semantic structure of the nerflets decomposition is beneficial even for lower level tasks, like novel view synthesis. We perform an experiment on the KITTI-360 validation set and observe that when training without a semantic or instance loss (i.e., photometric and regularization losses only), nerflets achieve a PSNR of 20.95. But when adding the semantics loss, PSNR *increases* to 22.43, because the nerflets end up more accurately positioned where the content of the scene is. This is also why we train with a higher semantic loss early in

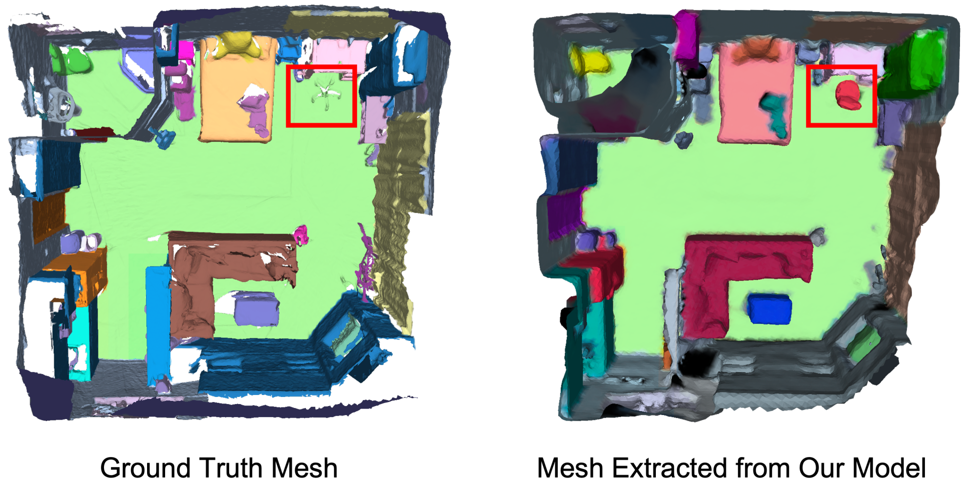

Figure 3.9. A 3D mesh extracted from a ScanNet RGB sequence with nerflets, colorized according to predicted panoptic labels. Ground truth RGBD mesh with human-annotated labels is shown on the left. Nerflets successfully reconstruct and label a chair instance missing from the ground truth mesh.

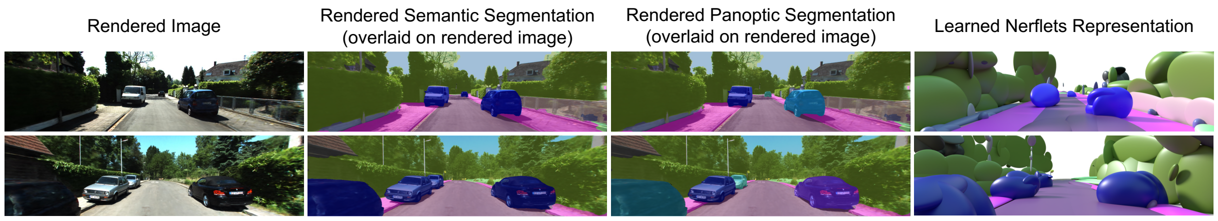
Rendered Image
Rendered Semantic Segmentation (overlaid on rendered image)
Rendered Panoptic Segmentation (overlaid on rendered image)
Learned Nerflets Representation

Figure 3.10. Visualization of nerflet outputs, trained on KITTI-360 images.

training, to encourage better nerflet positioning.

Ablation Study: Here we run a knock-out ablation study to validate the effectiveness of each of our regularization terms. Table 3.4 shows that all regularization terms contribute to final performance quantitatively. $\mathcal{L}_{\text{density}}$ is the most crucial term for learning a nice representation. It affects both image synthesis and segmentation performance, as it encourages nerflets to focus around actual scene content. $\mathcal{L}_{\text{radii}}$, $\mathcal{L}_{\ell_1}$ and $\mathcal{L}_{\text{box}}$ all also improve performance, due to their effect of forcing a more well-separated and active decomposition of the scene where all nerflets contribute to the final result.

Performance: Nerflets have good performance due to their local structure. Our editor renders 320×240 top-1 editable volume images with 192 samples/pixel at 31 FPS with 4

Table 3.4. Ablation experiment on ScanNet for the effectiveness of our regularization terms– density loss $\mathcal{L}_{\text{density}}$, radii penalty $\mathcal{L}_{\text{radii}}$, influence sparsity loss $\mathcal{L}_{\ell_1}$ and scene box loss. $\mathcal{L}_{\text{box}}$

	PSNR	mIOU	mAP0.5
w/o $\mathcal{L}_{\text{density}}$	20.85	63.31	11.20
w/o $\mathcal{L}_{\text{radii}}$	27.23	72.43	26.32
w/o $\mathcal{L}_{\ell_1}$	28.83	68.23	21.74
w/o $\mathcal{L}_{\text{box}}$	28.93	72.14	29.88
full model	**29.12**	**73.63**	**31.32**

Table 3.5. Ablation experiment on ScanNet for different number of nerflets.

	PSNR	mIOU
$n = 64$	26.34	53.23
$n = 128$	28.34	62.41
$n = 256$	28.81	69.97
$n = 512$	29.12	73.63
$n = 1024$	29.19	74.09

A100 GPUs and 64 nerflets– 457 million sample evaluations per second.

Number of Nerflets: We perform ablation study on the number of nerflets on the subset of ScanNet from [13]. The results are in Table 3.5. We find that increasing the number of nerflets could improve the performance on both photometric and semantic metrics. However, the benefit saturates when adding more nerflets than 512. To balance performance and efficiency, we use 512 nerflets in all experiments in the paper.

Effect of Top-k Evaluation: We perform ablation study on the performance impact of k when we only evaluate nerflets with top-k influence weights for each point sample. The experiment is done on the subset of ScanNet from [13]. The results are in Table 3.6. We

Table 3.6. Ablation experiment on ScanNet for evaluating only nerflets with top-k influence weights during training and testing.

	PSNR	mIOU
$k = 32$	29.13	73.72
$k = 16$	29.12	73.63
$k = 3$	29.05	72.95
$k = 1$	28.35	70.73

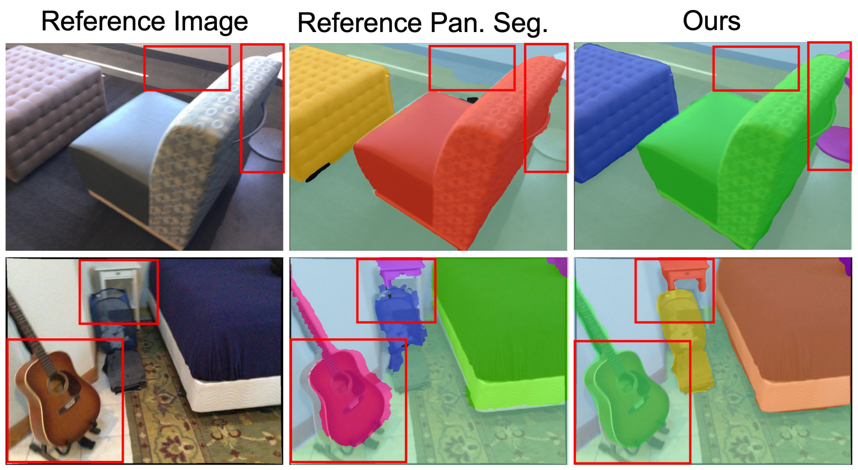

Figure 3.11. Comparison of ScanNet reference panoptic segmentation maps and our panoptic segmentation predictions overlaid on reference images.

find that evaluating 32, 16 or 3 nerflets have little influence on the model performance, since each nerflet is only contributing locally. However we see a moderate performance drop when only evaluating one nerflets with the highest influence weight. We choose to evaluate $k = 16$ nerflets for all experiments in the paper to balance the computational cost and performance.

Inactive Nerflets: One known problem [47] with training using RBFs that have learned extent is that when an RBF gets too small or too far from the scene, it does not contribute to the construction results. The radii loss $\mathcal{L}_{\text{radii}}$ and box loss $\mathcal{L}_{\text{box}}$ are proposed to alleviate this issue. To estimate the actual number of inactive nerflets, we utilize nerflet influence map W and count nerflets that do not appear on any of these maps in any views. In KITTI-360 experiments, we estimate to have 10.6 inactive nerflets on average per scene, making up $\sim 2.07\%$ of all available nerflets. In ScanNet experiments in the main paper, we estimate to have 30.6 inactive nerflets on average per scene, making up $\sim 5.98\%$ of all available nerflets.

Robustness against Input 2D Segmentation: In Figure 3.11, we visualize more examples on ScanNet comparing our panoptic predictions with reference annotations from the dataset. It can be seen that our representation learned from 2D supervision contains rich information and can produce more accurate segmentation results than reference maps in some cases. Our method produces clearer boundaries, fewer holes and discovers missing objects in the reference results, thanks to its ability to fuse segmentation maps from multiple views with a 3D sparsity prior from the structure of the representation.

3.7 Conclusion and Limitations

In this work, we present nerflets, a novel 3D scene representation which decomposes the scene into a set of local neural fields. Past work demonstrated structure is useful for parsimony in MLP-based shape representation [46], and we have found similar evidence extending that to scenes in this work. Thanks to the locality of each nerflet, our model is compact, efficient, and multi-view-consistent. Results of experiments on two challenging real-world datasets KITTI-360 and ScanNet demonstrate state-of-the-art performance for panoptic novel view synthesis, as well as competitive novel view synthesis and support for downstream tasks such as scene editing and 3D segmentation.

Despite these positives, nerflets have several limitations. For example, we do not model dynamic content. Even though the representation is well-suited for handling rigid motions (as demonstrated in scene editing), that feature has not been investigated. Also, while individual nerflet radiance fields are capable of handling participating media, the overall representation may struggle to fit scenes where those effects cross semantic boundaries (e.g., foggy outdoor sequences). Finally, we currently assume a fixed number of nerflets for each scene, regardless of the scene complexity. However, it may be advantageous to prune, add, or otherwise dynamically adjust the number based on where they are needed (*e.g.* where the loss is highest). Investigating these novel features is interesting for future work.

Connection to 3D Gaussian Splatting (3DGS): This work shares a common high-level idea with the later work 3DGS [78]. Both methods explore to decompose and express a large scene into a set of small 3D Gaussian based primitives. The major difference lies in the method for rendering. In our work, we follow the NeRF line of works to evaluate point samples along the ray, and render with volume rendering equation. While in 3DGS, splatting is used to directly rasterize 3D representations onto 2D image planes. Our representation is more information-rich while 3DGS is more compact and faster in rendering.

Acknowledgements

Chapter 3, in full, is a reprint of the material published in the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR): “Nerflets: Local Radiance Fields for Efficient Structure-Aware 3D Scene Representation from 2D Supervision” (Xiaoshuai Zhang, Abhijit Kundu, Thomas Funkhouser, Leonidas Guibas, Hao Su, and Kyle Genova). The dissertation author was the primary investigator and author of this paper. We sincerely thank Yueyu Hu, Nilesh Kulkarni, Songyou Peng, Mikaela Angelina Uy, Boxin Wang, and Guandao Yang for useful discussion and help on experiments. We also thank Avneesh Sud for feedback on the manuscript.

Chapter 4

Large-Scale Consistent 2D-3D Pre-Training with Dense and Sparse Features

4.1 Introduction

The rapid advancement of 3D computer vision has led to significant breakthroughs in understanding and interpreting the world in three dimensions. However, achieving robust performance across a range of 3D perception tasks is challenging when we try to match the accomplishments of large pre-trained models in the natural language and 2D vision domains. The path to a 3D foundation model is hampered by the relative scarcity of 3D data compared to 2D images, and especially the increased difficulty of obtaining quality annotations in 3D. At the same time, 3D models need to co-exist and communicate with language or language-vision models, if we are to optimally use priors in perceiving, reasoning, and acting on the physical world. Motivated by these considerations we propose a novel approach for large-scale 3D pre-training that leverages the knowledge encoded in extant pre-trained 2D networks, capitalizes on the availability of large-scale multi-view datasets, and learns consistent 2D-3D features.

In this work, we introduce a comprehensive 2D-3D joint training scheme, named ConDense, aimed at extracting co-embedded 2D and 3D features in an end-to-end

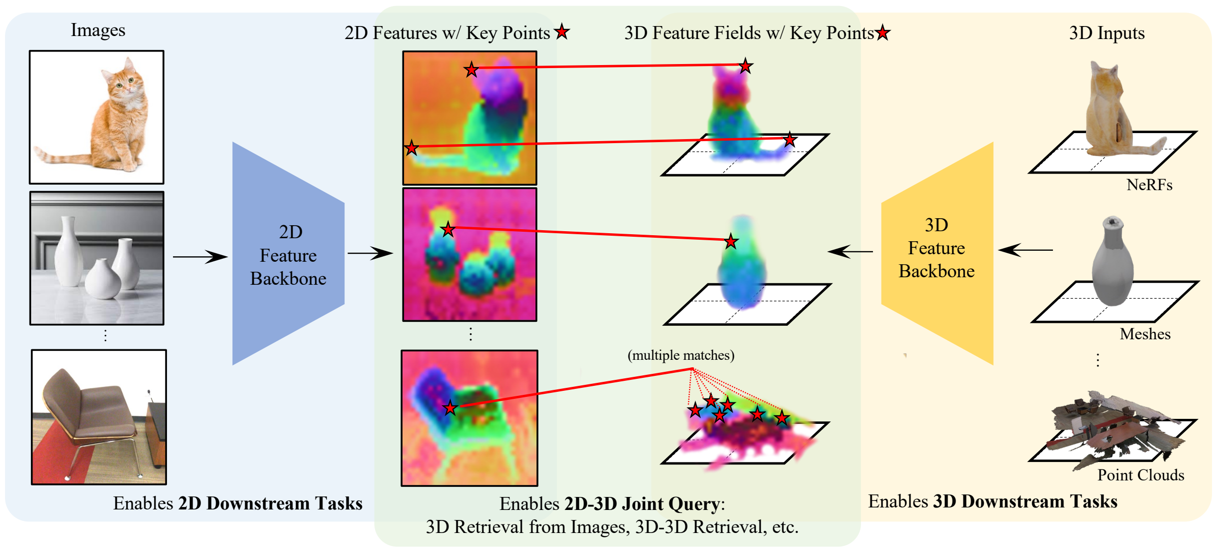
Images
2D Features w/ Key Points
3D Feature Fields w/ Key Points
3D Inputs
2D Feature Backbone
3D Feature Backbone
NeRFs
Meshes
Point Clouds
(multiple matches)
Enables **2D Downstream Tasks**
Enables **2D-3D Joint Query**:
3D Retrieval from Images, 3D-3D Retrieval, etc.
Enables **3D Downstream Tasks**

Figure 4.1. ConDense extract co-embedded feature for 2D or 3D inputs. The model not only has improved performance over previous pre-training methods but also enables efficient cross-modality, cross-scale queries such as 3D retrieval and duplicate detection.

pipeline. Our approach goes beyond conventional pre-training methods by enforcing 2D-3D feature consistency through 2D-3D consensus. This consistency is established by cross-checking the extracted 2D and 3D features via a ray-marching process inspired by Neural Radiance Fields (NeRFs), ensuring that the learned features align seamlessly in both the 2D and 3D domains.

In addition, ConDense represents the extracted features in two forms: a dense per-pixel representation and a sparse key point-based representation. This dual representation allows us to capitalize on the strengths of both types of features, making ConDense versatile and adaptable to a wide range of downstream tasks. Consider, for example, the fact the NeRFs have made the capture of 3D scenes a lightweight and widely available process. We will soon have hundreds of millions of objects and scenes in a NeRF form and the need arises to organize and search large collections of such data – and in particular to interrogate them using language and image queries. Our sparse key point representation and joint 2D-3D embeddings enable and facilitate such multi-modal cross-domain queries.

Our contributions can be summarized as follows:

- We propose ConDense, a novel 3D self-supervised pre-training scheme. By leveraging only large-scale 2D multi-view image datasets and 2D foundation models, we are able to achieve 3D pre-training with state-of-the-art downstream task performance, even when compared to 3D pre-training methods that utilize 3D training data.

- Our approach leads to the creation of more consistent and less noisy 2D features, enhancing the quality of existing 2D visual representations, and shows better performance than base models in various downstream 2D tasks.

- By learning sparse features jointly with the dense features for both 2D and 3D, we enable several novel tasks such as efficiently matching 2D images to larger-scale 3D scenes, or matching 3D captures of the same scene to each other.

- We establish a unified embedding space where 2D, 3D, and other modalities (e.g., natural language prompts) can be jointly queried, enabling efficient matching using either dense or efficient sparse features.

To validate the effectiveness of our large-scale pre-training approach, we conduct extensive experiments, showcasing its superior performance in various tasks. Furthermore, our pre-trained model opens up exciting possibilities for downstream applications, such as querying 3D scenes through natural language inputs or efficiently matching 2D images to 3D scenes, all without per-scene fine-tuning.

4.2 Related Work

2D Representation Learning and Foundation Models. Initial works on self-supervised 2D representation learning employ various pretext tasks derived from the images themselves [129, 221, 17], etc. Another line of work adopted discriminative strategies, such as instance classification [55], treating each image as a unique class and employing data augmentation for training. Recent advances in patch-based architectures, like Vision

Transformers (ViTs) [41], revived interest in pretext tasks, particularly inpainting in both image and feature space. Various works find that masked-autoencoders (MAEs) provide strong initialization for downstream tasks [54]. However, all these pre-trained features require additional supervised fine-tuning. More recently, foundation models, referring to pre-trained models adept at a broad range of tasks, have seen expansive growth within the vision domain through variants that have successfully adapted to numerous vision-related tasks. Notably, the CLIP model [139] leverages contrastive learning from extensive image-text pairs to achieve zero-shot task transferability. DINO [18, 123] demonstrates the emergence of various desirable properties in its features through self-supervision, facilitating its direct application across diverse visual tasks.

3D Representation Learning and Foundation Models. Despite advances in 2D representation learning and foundation models, 3D models lag behind greatly due to dataset and architecture constraints. A major line of research proposed various pretext tasks for 3D point clouds [126, 224]. The recent success of ViTs in 2D has also spurred the exploration of their counterparts in 3D domains [212, 126, 236, 95]. However, all these models require 3D point clouds for pre-training. Most of them are pre-trained on ScanNet [36] (around 1000 scenes) and ShapeNet [20] (around 50k objects, synthetic), and are thus constrained by the limited amount of *real-world* data available.

2D to 3D Feature Distillation and Multi-Modality Embeddings. With the recent development of large-scale 2D foundation models and multi-modality embeddings (*e.g.*, CLIP for images and languages), many have tried to distill the knowledge learned from these models and extend their application to 3D data formats. PointCLIP and PartSLIP [220, 102] achieves zero-shot point cloud classification and segmentation by projecting point clouds to 2D depth maps and applying the 2D pre-trained models directly. OpenShape, ULIP, and ULIP-2 [100, 199, 200] collects text, image, and point cloud triplets and takes advantage of pre-aligned vision-language feature space to achieve alignment among the triplet modalities. Specifically, they fix the visual-language embedding space and only tune their 3D point cloud encoder to achieve this alignment. These methods focus on the task of point cloud classification and their design cannot be easily extended to other 3D tasks or 3D input formats. Several methods have been proposed for dense feature encoding in 3D [130, 236, 232]. For example, OpenScene [130] trains on point cloud-grounded multi-view datasets and learns a 3D point cloud network from multi-view aggregated 2D features. Like PointCLIP and ULIP, OpenScene requires 3D point clouds for training, which are scarce and hard to collect in the real world on a large scale, when compared to multi-view images. These works also re-use the embedding space from the 2D foundation model and only distill the 3D encoder.

More recently, Neural Radiance Fields (NeRFs) [116] and numerous subsequent follow-ups [9, 222] have gained great success in novel view synthesis. NeRF has the property of aggregating information across views. Several recent works leverage this property to improve the quality of semantic segmentation [79, 223, 229]. Many works distill features (*e.g.*, DINO [18] and CLIP [140]) into 3D and demonstrate they can be used for downstream tasks such as natural language-based query. These works require per-scene distillation and optimization. FeatureNeRF [207] proposed to distill features from 2D foundation models to 3D space via generalizable NeRFs [209, 23]. Through distillation, the learned model can lift any 2D images to continuous 3D semantic feature volumes. However, the pipeline serves more as a 2D-to-3D lifting technique and cannot handle native 3D data directly.

NeRF for Perception. The integration of Neural Radiance Fields (NeRF) into various discriminative perception tasks such as classification, detection, and segmentation has also been explored [223, 174]. These methods typically follow a reconstruction-then-detection pipeline by creating NeRFs from multi-view image data first and then designing task-specific networks and loss terms to tackle each perception task, and many of them work in a scene-by-scene manner and require re-training and optimization for each new scene. More recently, NeRF-Det [195] incorporates generalizable, feature-conditioned NeRF, and 3D detection pipelines to achieve efficient detection performance with no per-scene fine-tuning. Our work is inspired by these ideas while developing further by joining forces with 2D foundation models and creating a pre-training pipeline that is native to both 2D images and 3D formats.

Querying 3D Data. A variety of works have explored 3D to 3D similarity queries in pre-deep learning period, mostly at the object level, by constructing human-designed whole object features encoded as Euclidean embeddings or as distributions [124, 24]. Later some works explored learned embedding spaces, as well as co-embeddings of images and 3D models [91]. Inspired by the bag-of-words paradigm in image search, "bags-of-features" have also been investigated in 3D to 3D search, for example [16]. However, none of these approaches is integrated into a multi-task framework, as we aim to do here and they are largely focused on object retrieval, not scenes.

4.3 Preliminaries

Neural Radiance Fields (NeRFs) [116] offer a novel representation of 3D scenes, capturing continuous volumetric scenes as neural networks. We briefly describe the mechanism of the NeRF and refer to [116, 9] for details about related NeRF models. A NeRF $\mathcal{F}$ maps a 3D coordinate $\mathbf{x} = (x, y, z)$ and a viewing direction $\mathbf{d} = (\theta, \phi)$ to a color $\mathbf{c} = (R, G, B)$ and density σ – $\mathcal{F} : (\mathbf{x}, \mathbf{d}) \mapsto (\mathbf{c}, \sigma)$, where color $\mathbf{c}$ is related to both point location $\mathbf{x}$ and viewing direction $\mathbf{d}$, recording the local appearance information, and density σ is only related to point location $\mathbf{x}$, recording the local geometry information.

The rendering of a 3D scene from a 2D perspective is formulated as a volume rendering problem. Given a ray $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$, where $\mathbf{o}$ is the camera origin and t is the distance along the viewing direction $\mathbf{d}$, the color $\mathbf{C}(\mathbf{r})$ of the ray is computed as:

$$\mathbf{C}(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\sigma(\mathbf{r}(t))\mathbf{c}(\mathbf{r}(t), \mathbf{d})\, dt, \;\; T(t) = \exp\left(-\int_{t_n}^{t} \sigma(\mathbf{r}(s))\, ds\right), \tag{4.1}$$

where $T(t)$ is the accumulated transmittance along the ray from t_n (near bound) to t_f (far bound). In practice, discretized approximations are used to evaluate this integral. The process is also called ray marching, which can be used as a general tool to render from any 3D feature field $\mathbf{f}$ and get a 2D-projected feature map as shown in many previous works [229, 174, 223]. The property can be used to bridge the 3D feature field of a scene with its 2D feature maps and is the basis of our 2D-3D consensus pipeline.

2D Foundation Models are deep learning models that have been extensively pre-trained on large-scale image datasets. Let $\mathcal{G}$ be a 2D foundation model that maps an input image $\mathbf{I}$ to a feature representation $\mathbf{F}$: $\mathcal{G} : \mathbf{I} \mapsto \mathbf{F}$. The feature representation $\mathbf{F}$ is a high-dimensional vector that captures the essential properties such as geometry and semantic information of the input image $\mathbf{I}$. Of particular interest, our work leverages DINO [18, 123], a self-supervised learning approach for visual representation trained on large-scale image datasets. Its latest version, DINOv2, excels in capturing both fine-grained details and global contextual information from images without any labeled data. By initializing the 2D encoder from DINOv2, we tap into a robust source of information-rich 2D dense features, and can thus kick-start our 3D encoder from 2D-3D knowledge distillation.

4.4 Method

An overview of our approach is illustrated in Figure 4.2 (dense feature encoding) and Figure 4.3 (key point prediction). Our model is composed of two branches, encoding 2D and 3D information, respectively (Sec. 4.4.1 and Sec. 4.4.2). Both branches encode features in two forms – dense format and key-point-based sparse format (Sec. 4.4.4). During training, we use paired 2D-3D inputs in the form of multi-view images and the corresponding NeRF scene. The information extracted in 2D and 3D branches is compared via 2D-3D consensus (Sec. 4.4.3) so that information can flow both ways.

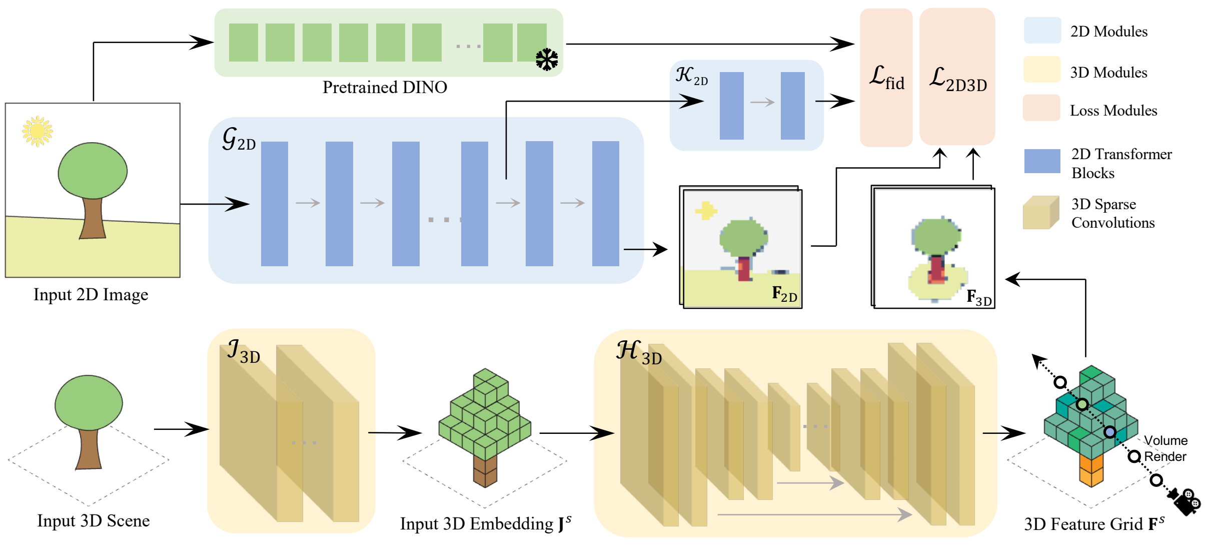

Figure 4.2. Dense feature encoding: the 3D encoding module $\mathcal{G}_{3D}$ is composed of a swappable input processing head $\mathcal{J}_{3D}$ and a common 3D reasoning backbone $\mathcal{H}_{3D}$. $\mathcal{J}_{3D}$ maps input 3D scenes of various formats into a feature $\mathbf{J}^s$ in a unified 3D embedding space. $\mathcal{H}_{3D}$ turns $\mathbf{J}^s$ into a 3D feature grid $\mathbf{F}^s$. Through interpolation on $\mathbf{F}^s$ and volume rendering, a 3D-projected feature map $\mathbf{F}_{3D}$ can be obtained and compared with a 2D dense feature map $\mathbf{F}_{2D}$, extracted from the 2D encoding module $\mathcal{G}_{2D}$. The resulting 2D-3D consensus loss $\mathcal{L}_{2D3D}$ is used as a self-supervision signal. An additional 2D fidelity loss $\mathcal{L}_{fid}$ is introduced to make sure that the 2D-3D consensus optimized 2D feature $\mathbf{F}_{2D}$ does not deviate too much from the original 2D feature in order to retain some of its semantics and visual richness.

4.4.1 2D Encoding

The 2D encoding branch ($\mathcal{G}_{2D}$) of the ConDense framework is crucial for extracting rich visual features from multi-view images, which later are learned synergistically with the 3D encoding module. We follow the network architecture of DINOv2 [123] to use ViT [41] as the base for our 2D encoder and load the pre-trained DINO weights for the initialization of our 2D branch. The ViT architecture takes as input a grid of non-overlapping contiguous image patches of resolution $N \times N$. In this work, we use $N = 14$ ("/14" ViT models). The patches are then passed through a linear layer to form a set of embeddings. Following previous works [41, 18], an extra learnable `[CLS]` token is added to the sequence to aggregate global information. These patch tokens and the `[CLS]` token are fed to the standard transformer blocks and updated by the attention mechanism.

For any given input image $\mathbf{I}$, the 2D branch generates a dense feature map $\mathbf{F}_{\texttt{2D}} = \mathcal{G}_{\texttt{2D}}(\mathbf{I})$. This feature representation $\mathbf{F}_{\texttt{2D}}$ is a high-dimensional vector encoding various essential attributions of the input image, and can already be used out-of-the-box due to its pre-training on large-scale image datasets. We will later show that combined with our 2D-3D joint training, the feature branch can be further improved for various downstream tasks.

4.4.2 3D Encoding

The 3D encoding branch ($\mathcal{G}_{\texttt{3D}}$) of the ConDense framework is aimed at extracting a 3D feature field from various data formats. It is designed to be composed of two parts: $\mathcal{G}_{\texttt{3D}} = \mathcal{H}_{\texttt{3D}} \circ \mathcal{J}_{\texttt{3D}}$, where $\mathcal{J}_{\texttt{3D}}$ is the input processing head and $\mathcal{H}_{\texttt{3D}}$ is the actual 3D reasoning backbone. Different 3D data formats have their individual input processing heads, but they share a common 3D reasoning backbone $\mathcal{H}_{\texttt{3D}}$. The major 3D inputs we are dealing with are NeRF models, while we also support other data formats such as point clouds. Here we detail the full pipeline for 3D feature field reasoning from NeRF data.

Grid Sampling from NeRF. Given any learned NeRF function $\mathcal{F} : (\sigma, \mathbf{c})$, we sample uniformly on a 3D lattice within the normalized scene bounding box $[-1, +1]$, with spacing ϵ between samples:

$$\Sigma^{\mathrm{s}} = \{\sigma(x), \text{s.t. } \mathbf{x} \in [-1 : \epsilon : 1]^3\}, \tag{4.2}$$

$$\mathbf{C}^{\mathrm{s}} = \{\mathbf{c}(x, \mathbf{d}), \text{s.t. } \mathbf{x} \in [-1 : \epsilon : 1]^3, \mathbf{d} \in \mathcal{D}\}, \tag{4.3}$$

where $\mathcal{D}$ is a predefined set of directions aiming to capture as much local appearance information as possible. In order to reduce computation costs, we use the density field from the input NeRF to sparsify these grids. Specifically, we calculate sample opacity by $\alpha(\mathbf{x}) = 1 - \exp(-\sigma(\mathbf{x}))$ due to the canceled-out spacing term δ_t in the regular grid [195], and filter out the point samples $\mathbf{x}$ with $\alpha(\mathbf{x}) < \theta$. After sparsification, the evaluated sigma value and color values are concatenated for each grid sample and fed into the input processing head, which is a small 3D sparse convolutional network:

$$\mathbf{J}^s = \mathcal{J}_{\texttt{3D}}^{\text{NeRF}}(\texttt{Concat}(\Sigma^{\text{s}}, \mathbf{C}^{\text{s}})). \tag{4.4}$$

Here $\mathbf{J}^s$ serves as the input embedding for the 3D reasoning backbone, while different input processing heads take care of mapping data from different input sources to this input embedding space. For point cloud inputs, we first voxelize the data before feeding it into a small 3D network. Please check the appendix for more details.

3D Spatial Reasoning. We apply a 3D UNet [35] implemented with sparse convolution blocks to obtain a 3D feature grid $\mathbf{F}^s$ from the aforementioned input embedding:

$$\mathbf{F}^{\text{s}} = \mathcal{H}_{\texttt{3D}}(\mathbf{J}^s). \tag{4.5}$$

The module enables reasoning in 3D space. To obtain the feature for an arbitrary 3D query point $\mathbf{x} \in \mathbb{R}^3$, we interpolate within the feature grid $\mathbf{F}^s$ with a trilinear interpolation operator:

$$\mathbf{f}_{\texttt{3D}}(\mathbf{x}) = \texttt{TriLerp}(\mathbf{x}, \mathbf{F}^s). \tag{4.6}$$

4.4.3 2D-3D Consensus with 2D Fidelity

With the proposed 2D and 3D encoders, our model can generate co-embedded dense features given 2D or 3D inputs. During training, we use paired data of multi-view 2D images $\{\mathbf{I}_k\}$ and their corresponding learned NeRF $\mathcal{F} : (\sigma, \mathbf{c})$ to jointly train the 2D and 3D branches.

Specifically, for each scene, we first generate the 3D feature field $\mathbf{f}_{\texttt{3D}}$ as detailed in

Sec. 4.4.2. Based on this feature field and the scene density σ, we can adapt the rendering equation to render 3D-projected feature maps as in [229, 174]:

$$\mathbf{F}_{\texttt{3D}}(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\sigma(\mathbf{r}(t))\mathbf{f}_{\texttt{3D}}(\mathbf{r}(t))\,dt. \tag{4.7}$$

In the meantime, we generate the 2D feature map with our 2D branch: $\mathbf{F}_{\texttt{2D}} = \mathcal{G}_{\texttt{2D}}(\mathbf{I}_k)$ and adopt the consistency loss between the 3D rendered feature map $\mathbf{F}_{\texttt{3D}}$ the 2D-originated counterpart $\mathbf{F}_{\texttt{2D}}$ with L_2 loss:

$$\mathcal{L}_{\texttt{2D3D}}(\mathbf{F}_{\texttt{2D}}, \mathbf{F}_{\texttt{3D}}) = \sum_{\mathbf{r}\in\mathcal{R}} ||\mathbf{F}_{\texttt{2D}}(\mathbf{r}) - \mathbf{F}_{\texttt{3D}}(\mathbf{r})||_2^2. \tag{4.8}$$

$\mathcal{R}$ denotes all camera rays in the multi-view image set of the scene that hit at least one active voxel in the 3D feature grid $\mathbf{F}^s$. The loss encourages information to flow both ways – the 3D branch could learn to generate useful 3D feature fields from the 2D multi-view supervision, and the 2D branch could also benefit from consistent underlying 3D geometry and learn to extract less noisy, multi-view consistent, and 3D-informed features.

Due to the biased and scarcer nature of the existing multi-view image datasets, if we optimize the networks based only on this 2D-3D consistency loss, the feature quality may degrade due to trivial solutions and biased data distribution. To prevent this, we propose to insert an additional output head called 2D fidelity head $\mathcal{K}_{\texttt{2D}}$ before the second-to-last transfer block (see Figure 4.2), and apply the 2D fidelity loss to keep its output $\mathbf{F}_{\texttt{2D}}^{\texttt{fid}}$ from deviating too much from the original DINOv2 [18] feature:

$$\mathcal{L}_{\texttt{fid}}(\mathbf{F}_{\texttt{2D}}^{\texttt{fid}}) = ||\mathbf{F}_{\texttt{2D}}^{\texttt{fid}} - \mathbf{F}_{\texttt{2D}}^{\texttt{t}}||_2^2, \tag{4.9}$$

where $\mathbf{F}_{\texttt{2D}}^{\texttt{t}}$ is the pre-trained DINOv2 feature. No ground-truth labels are used in this loss, and this term can be applied on any natural image collection. We adapt ImageNet-21k [144] in our pipeline.

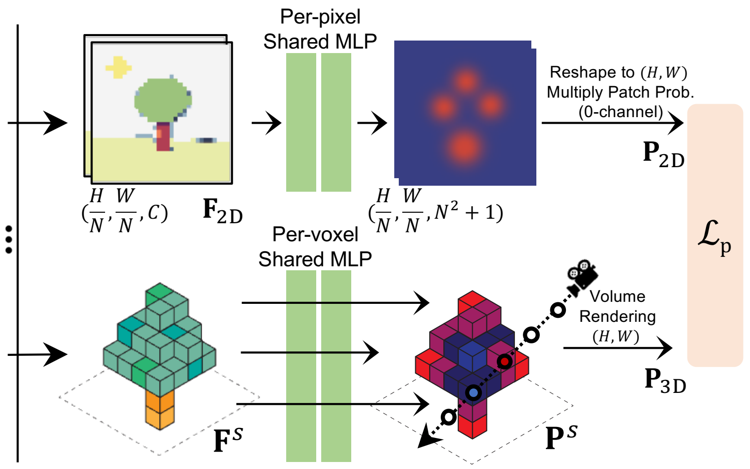

Figure 4.3. Key point prediction: key points are detected in both 2D ($\mathbf{P}_{\text{2D}}$) and 3D ($\mathbf{P}_{\text{3D}}$) based on the existing feature backbones. The 2D-3D key point loss $\mathcal{L}_\text{p}$ is used as a self-supervision signal.

4.4.4 Key Point Extraction

With the proposed 2D and 3D encoders and the joint training scheme, our model can generate co-embedded dense features given any input 2D image or 3D scene. This property is desired since it enables possible applications to query among 2D, 3D, and other modalities. To further facilitate these applications, we add support for sparse key point detection in both 2D and 3D based on the existing feature backbones for efficient queries across scene scales.

As shown in Figure 4.3. To detect key points in 2D images, we follow a similar formulation as in [40] and decode the 2D backbone feature $\mathbf{F}_{\text{2D}}$ into the full image-resolution interest point possibility map $\mathbf{P}_{\text{2D}}$ with 2 MLP layers with a softmax output head (noted as $\mathcal{M}_{\text{2D}}$). Please refer to [40] and our Appendix for more in-depth details. For the 3D branch, we similarly use 2 MLP layers with a ReLU output head (noted as $\mathcal{M}_{\text{3D}}$) to decode the key point possibility on the 3D feature grid $\mathbf{F}^s$, and the possibility maps are rendered and compared between 2D and 3D using the aforementioned 2D-3D consensus scheme:

$$\mathbf{P}_{\texttt{2D}} = \mathcal{M}_{\texttt{2D}}(\mathbf{F}_{\texttt{2D}}), \tag{4.10}$$

$$\mathbf{P}_{\texttt{3D}}(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\sigma(\mathbf{r}(t))p_{\texttt{3D}}(\mathbf{r}(t))\,dt, \tag{4.11}$$

$$\mathbf{P}^s = \mathcal{M}_{\texttt{3D}}(\mathbf{F}^s),\ p_{\texttt{3D}}(\mathbf{x}) = \texttt{TriLerp}(\mathbf{x}, \mathbf{P}^s), \tag{4.12}$$

$$\mathcal{L}_{\texttt{p}}(\mathbf{P}_{\texttt{2D}}, \mathbf{P}_{\texttt{3D}}) = \sum_{\mathbf{r}\in\mathcal{R}} ||\mathbf{P}_{\texttt{2D}}(\mathbf{r}) - \mathbf{P}_{\texttt{3D}}(\mathbf{r})||_2^2. \tag{4.13}$$

During test time, 3D key points are selected directly from opaque 3D grid samples. We multiply the opacity value by the predicted key point possibility and use $\alpha(x) \times p_{\texttt{3D}}$ as the selecting criteria for the 3D key points.

We argue that the joint 2D-3D key point detection not only helps enable various query tasks based on dense feature backbones (Sec. 4.6.3) but also serves as a useful technique to improve the overall feature quality (Sec. 4.7.1).

4.5 Implementation Details

Loss Terms. The final loss $\mathcal{L}$ is given by:

$$\mathcal{L} = \lambda_{\texttt{2D3D}}\mathcal{L}_{\texttt{2D3D}} + \lambda_{\texttt{fid}}\mathcal{L}_{\texttt{fid}} + \lambda_{\texttt{p}}\mathcal{L}_{\texttt{p}}, \tag{4.14}$$

where $\lambda_{\texttt{2D3D}}$, $\lambda_{\texttt{fid}}$, and $\lambda_{\texttt{p}}$ are scalars adjusted throughout the training process. Check the appendix for more details.

Datasets Details. For all experiments included in the main experiments, we use MVImgNet [211], ScanNet [36], and RealEstate10k [234] as our multi-view pre-training datasets. MVImgNet is an object-centric dataset containing a total of 6.5 million frames from more than 200k video captures of diverse objects. ScanNet and RealEstate10k are indoor scene-scale datasets each containing a diverse set of scene captures in the form of video clips. Though other modalities are provided in some of these datasets (*e.g.* point clouds, semantic labels, etc.), we only use the posed images in pre-training. To generate NeRFs for our pre-training dataset, we use the MipNeRF-360 [9] official implementation to fit the scenes. For MVImgNet, we trained 4000 steps for each scene. For ScanNet and RealEstate10k we trained 8000 steps for each scene. We half the image resolution before using it as training supervision. All the NeRF are fitted with 8 V100 GPUs. We find these settings are enough for generating NeRF with good qualities. We summarized the NeRF quality in terms of PSNR and SSIM in Table 4.1. It takes around 35000 V100 GPU hours to build these NeRFs, and we parallelize the process on around 1000 GPUs. For ScanNet, we are determining the bounding boxes with the ground-truth meshes. For MVImgNet and RealEstate10k, we use the ray near-far values and camera locations to determine the scene bounds.

Table 4.1. NeRF quality on the pre-training datasets with PSNR (before the slash) and SSIM (after the slash), tested on the sampled 1% images on each dataset's native image resolution.

MVImgNet	ScanNet	RealEstate10k
30.23 / 0.939	25.78 / 0.901	27.24 / 0.922

We acknowledge that fitting NeRFs on multi-view datasets requires significant computation. However, we believe this upfront cost is justified: (1) It enables 3D backbone pre-training without explicit 3D supervision, which is much more *time-consuming* to acquire. (2) Creating these datasets is a one-time process, and they can be shared among researchers to avoid repeated computation. (3) Faster rendering models have emerged since we developed the pipeline, especially the 3D Gaussian Splatting, which is even more suitable for our pipeline due to its sparsity nature. Adoption of these newer models could potentially cut the computational cost greatly.

ImageNet-21k [144] is used to provide image samples for the 2D fidelity loss in addition to the multi-view datasets. ImageNet-21k is the superset of the commonly used ImageNet-1k dataset and contains more than 14 million images.

2D Input Details. For MVImgNet [211], we use all 6.4M images in 214k scenes spreading over 238 classes for pre-training. Images are center-cropped and resized to 336×336 before being fed into 2D branch input. For ScanNet [36], we use all 1513 scans for training. We filter out blurred frames by calculating the variance of the Laplacian matrix and ignore them for the 2D branch inputs. Other images are randomly cropped in 336×336 before being used as 2D inputs. The same sampling and pre-processing schemes are applied for RealEstate10k [234]. For 2D fidelity loss, we sample from all 14M images in ImageNet-21k spreading over 21k classes. Images are resized in 336×336 before being processed by our 2D branch and output features were compared with the DINOv2 ViT-g/14 model [123] to enforce the 2D fidelity.

3D Input Details. During training, we grid sample the NeRFs with resolution varying from $64 \times 64 \times 64$ to $256 \times 256 \times 256$ to ensure adaptability to different resolutions. These samples are then re-scaled to $128 \times 128 \times 128$, with 0.5 possibility of being sparse-dilated. The output 3D feature grid $\mathbf{F}^s$ has the spatial resolution of $128 \times 128 \times 128$ in all experiments in this work.

Network Details. For the 3D Network, we follow the conventions and implementations as in MinkowskiNet [34]. Specifically, for input processing heads $\mathcal{J}_{\mathtt{3D}}$, we apply 3 sparse convolution layers with "$5 \times 5 \times 5 \times 1, 16$", "$5 \times 5 \times 5 \times 1, 32$" and "$5 \times 5 \times 5 \times 1, 64$" configurations, similarly following the input processing of [34]. Here $\times$ indicates a hypercubic sparse kernel. For the 3D reasoning backbone $\mathcal{H}_{\mathtt{3D}}$, we apply the same architecture as MinkowskiUNet32 in [34] but remove the original input head and modify the number of output channels to match the 2D feature channels. We utilize Vision Transformers (ViT-g/14) as the backbone for the 2D branch and use 8 A100 GPUs for training. For the key point prediction, the process is illustrated in Figure 4.3. Two different 2-layer MLPs are used for reasoning key point possibilities from 2D and 3D inputs.

Training Scheme. We bootstrap the full training process of ConDense in four stages. First, the 2D feature backbone $\mathcal{G}_{\mathtt{2D}}$ is initialized from DINOv2 [18] pre-trained weights.

We then freeze its weights and fit the 2D key point detector MLPs ($\mathcal{M}_{\texttt{2D}}$) by enforcing the interest point heatmap predictions to a pre-trained, frozen SuperPoint [40] model. Then both $\mathcal{G}_{\texttt{2D}}$ and $\mathcal{M}_{\texttt{2D}}$ are kept frozen, while the 3D branch modules $\mathcal{G}_{\texttt{3D}}$ and $\mathcal{M}_{\texttt{3D}}$ are optimized from $\mathcal{L}_{\texttt{2D3D}}$ and $\mathcal{L}_{\texttt{p}}$. In this stage, we distill the knowledge from the 2D foundation models to kick start the learning of 3D modules. For the final phase, we unfreeze all modules and jointly train all 2D and 3D modules with the loss terms defined in Equation 4.14.

Loss Scheduling. After initializing the 2D branches and fine-tuning the 2D key point detector, we train for 30k iterations with $\lambda_{\texttt{fid}}$ and $\lambda_{\texttt{p}} = 0$ while only tuning the 3D networks (distillation from 2D to 3D), where we sample 8 NeRF scenes in each iteration and sample rays within the original set of rays for supervision. After this stage, we train an additional 30k iterations with a linear warm-up of $\lambda_{\texttt{fid}}$ and $\lambda_{\texttt{p}}$ in 5k iterations where all 2D and 3D modules are jointly fine-tuned. Throughout the process, we use an AdamW optimizer, an initial LayerScale value of 1e-5, a weight decay cosine schedule from 0.02 to 0.24, a learning rate of 3.3e-4, and its warm-up of 2k iterations.

Computational Footprints. For data preparation, we use V100 GPUs for fitting each scene. It takes around 35k V100 GPU hours. For pre-training, we use A100 GPUs and it takes around 4k A100 GPU hours, including both distillation and joint tuning stages. The total estimated power consumption is 12.1 MWh and carbon emitted is 5.8t CO2eq.

4.6 Experiments

In this section, we present extensive evaluations of our models on 1) 3D tasks including 3D classification, and 3D segmentation; 2) 2D image understanding tasks; and 3) Cross-Modality scene queries. In all experiments, we freeze weights for the feature backbones $\mathcal{G}_{\texttt{2D}}$ and $\mathcal{H}_{\texttt{3D}}$ unless otherwise stated. Due to the page limit, we only include the most common benchmarks in the main paper, please check the appendix for more experiments including 3D detection, 2D retrieval, and 2D depth estimation.

4.6.1 3D Tasks

For point-cloud-based 3D tasks, we use the 3D feature backbone $\mathcal{H}_{\texttt{3D}}$ *out-of-the-box* and freeze its weights, while training a point cloud input head $\mathcal{J}_{\texttt{3D}}^{\text{PC}}$ with 4 sparse convolutional layers. For a 3D point $\mathbf{x}$, we fetch the interpolated features in the 3D feature grid $\mathbf{F}^s$ as the point feature (Equation 4.6). Depending on the actual 3D tasks, different output heads could be added to further process these point features.

3D Classification We follow the testing protocols in the previous works [199, 200] to evaluate on ModelNet40 [192] and ScanObjectNN [170]. ModelNet40 is a synthetic dataset of CAD models containing around 10k training samples and 2.5k testing samples. ScanObjectNN is a real-world 3D dataset with around 15k objects extracted from indoor scans. We follow the same dataset setup and preparation protocols as in ULIP [199] to ensure consistent evaluation. We apply normalization on the point clouds before passing them into the input processing head and use a simple 3-layer sparse convolutional network with average pooling and 1-layer MLP output with softmax to predict the scene class. Only these two modules are trained with standard cross entropy loss on the target dataset and the 3D feature backbone is kept frozen. Results are in Table 4.2.

3D Segmentation We follow the testing protocols in the previous works [204, 138] to evaluate on the ScanNet [36] and the S3DIS [6] datasets. ScanNet contains 1613 indoor scans with 20 semantic classes, we train on its train split and report the mean Intersection over Union (mIoU) on the validation split. S3DIS contains 272 scenes, we train on its training split and evaluate on its validation set with the 6-fold cross-validation scheme. The voxel sizes for ScanNet and S3DIS are set to $2\texttt{cm}$ and $5\texttt{cm}$, respectively. We extract the point feature from the backbone with trilinear interpolation (Equation 4.6) and use a simple linear layer with softmax to predict the point label. Only the input processing head and the linear layer are trained with standard cross entropy loss on the target dataset and

Table 4.2. 3D classification results on ScanObjectNN (before slash) and ModelNet40 (after slash). ConDense outperforms all the baselines including train-from-scratch methods and pre-training methods.

Model	Overall Acc	Cls-mean Acc
PointNet [135]	68.2 / 89.2	63.4 / 86.0
PointNet++ [136]	77.9 / 90.7	75.4 / – –
DGCNN [186]	78.1 / 92.9	73.6 / 90.2
MVTN [51]	82.8 / 93.8	– – / 92.0
PointMLP [112]	85.7 / 94.1	84.4 / 91.3
PointNeXt [138]	87.5 / – –	85.9 / – –
Point2Vec [213]	87.5 / 94.8	86.0 / 92.0
ULIP (w/ PointMLP) [199]	88.8 / 94.3	87.8 / 92.3
ULIP-2 (w/ PointNeXt) [200]	90.8 / – –	90.3 / – –
ReCon [137]	90.6 / 94.7	– – / – –
PointGPT [26]	93.4 / 94.9	– – / – –
ConDense (Ours)	**94.1 / 95.2**	**93.4 / 93.1**

the 3D feature backbone is kept frozen. Results are presented in Table 4.3.

3D Detection We show that our 3D features are also useful for 3D detection by following the settings of [204] to attach a state-of-the-art detection head CAGoup3D [178] and fine-tune the entire network for fair comparisons. The results are presented in Table 4.4. We provide an additional 2.1 and 2.9 points gain in terms of mAP@0.25 and mAP@0.5 respectively over an already strong baseline CAGroup3D. Our performance is also better than other pre-training methods [194, 204].

Summarizing the results in Table 4.2, Table 4.3, and Table 4.4 our method has demonstrated superior performance compared to other train-from-scratch and pre-training frameworks on 3D classification, segmentation, and detection tasks. Despite only tuning the input and output heads, our approach still surpasses the performance of pre-trained methods that fine-tune the entire model. Furthermore, while many other methods heavily utilize point cloud data during pre-training [199, 200], our approach achieves remarkable results without this requirement.

Table 4.3. 3D segmentation results (mIOU) on ScanNet and S3DIS. ConDense outperforms all the baselines including train-from-scratch methods and pre-training methods.

Model	ScanNet	S3DIS
PointNet++ [136]	53.5	54.5
MinkowskiNet [34]	72.2	65.4
PointCNN [90]	–	65.4
KPConv [168]	69.2	70.6
PointNeXt [138]	71.5	74.9
PointMetaBase [95]	72.8	77.0
PointVector [39]	–	78.4
PointContrast [194]	74.1	–
MSC (w/ SparseUNet) [191]	75.5	–
PPT (w/ SparseUNet) [190]	76.4	78.1
PonderV2 (w/ SparseUNet) [235]	77.0	79.9
Swin3D [204]	77.5	79.8
ConDense (Ours)	**79.8**	**80.7**

Table 4.4. 3D detection results (mAP@0.25 and mAP@0.5) on ScanNet.

Model	mAP@0.25	mAP@0.5
RepSurf [141]	71.2	54.8
SoftGroup [175]	71.6	59.4
CAGroup3D [178]	75.1	61.3
PointContrast [194]	59.2	37.3
Swin3D [204] w/ CAGroup3D	76.4	63.2
ConDense w/ CAGroup3D	**77.2**	**64.2**

4.6.2 2D Tasks

To evaluate the performance of the pre-trained 2D feature backbone $\mathcal{G}_{\texttt{2D}}$, we follow the settings as presented in DINOv2 [123], and compare with common self-supervised pre-trained baselines including MAE [54], DINO [18, 123], and iBOT [231], as well as weakly supervised visual-language pre-trained model OpenCLIP [71]. For both classification and segmentation, we present results under the "Linear (lin.)" setting [18, 123]. We include more results under the "multi-scale (+ms)" setting and more 3D benchmarks in our appendix. Results are presented in Table 4.5.

2D Classification We test the quality of the holistic image representation produced by the model on the ImageNet-1k [149] and Places205 [230] classification dataset. A linear probe is added on top of the frozen feature backbone to generate the prediction, following previous works [18, 123].

2D Segmentation We test on the task of semantic image segmentation to evaluate the quality of our learned representation. A linear layer is trained to predict class logits from patch tokens and it is upscaled to obtain the final segmentation map, following previous works [18, 123].

Table 4.5. 2D classification and segmentation results on multiple evaluation datasets with frozen features. ConDense improves over the DINOv2 in all benchmarks.

	Classification (Acc)		Segmentation (mIOU)	
Model	ImageNet	Places205	ADE20k	PascalVOC
OpenCLIP [71]	86.2	69.8	39.3	71.4
MAE [54]	76.6	52.4	33.3	67.6
DINO [18]	79.2	60.4	31.8	66.4
iBOT [231]	82.3	64.4	44.6	82.3
DINOv2 [123]	86.5	67.5	49.0	83.0
ConDense	**89.6**	**70.2**	**53.6**	**85.1**

For both 2D classification and segmentation benchmarks, our 3D-informed ConDense shows consistent improvement over the original DINOv2 and indicate that our 2D-3D consensus training pipeline can help improve the performance of the existing 2D foundation models.

2D Retrieval We evaluate our performance for image retrieval with MVImgNet [211] and ScanNet [36]. We follow the experiment settings of [18, 123] by freezing the features and directly applying k-NN for retrieval. On both datasets, we perform tests with 1 query image and 1 index image on each scene. The top-1 accuracy is reported in Table 4.6. Our ConDense clearly generates better global features suitable for retrieval tasks.

Depth Estimation We follow the settings in [123] and attach a linear classifier on top of one (lin. 1) or four (lin. 4) transformer layers to infer depth from the frozen feature backbones. It can be seen that our performance is better than all other pre-training backbones except DINOv2. The reason is that the 2D-3D consensus loss enforces the feature to be invariant across different views, and the 2D branch is thus expected to generate the same features for the same spatial point regardless of viewing angles and distances. The process of 2D-3D consensus facilitates more 3D-informed and consistent features, as can be observed from Figure 4.4. Such a property could be desired or unwanted depending on the exact downstream tasks. And we defer the more in-depth study into this property to future research. To validate this, we show our results on multi-view stereo (MVS) depth estimation. Here we tested two widely used MVS depth estimation methods– MVSNet [205] and PointMVS [28], on the standard dataset (DTU [73]). It can be seen that both backbones are boosted by our features, and ConDense outperforms DINOv2 by a large margin.

Table 4.6. 2D retrieval results (top-1 Acc) on MVImgNet and ReaEstate10k datasets.

Model	MVImgNet	RealEstate10k
OpenCLIP [71]	70.1	63.0
MAE [54]	65.4	55.1
DINO [18]	65.6	58.9
DINOv2 [123]	70.1	61.3
ConDense	**76.9**	**63.2**

4.6.3 Cross-Modality Scene Query

Leveraging the joint 2D-3D co-embedding property of ConDense, we are able to query across modalities. Here we tackle a series of matching tasks including scene identification from 2D images and a newly proposed task – 3D scene duplication detection. The results are presented in Table 4.9.

We use 3 datasets: Objectron [170], ScanNet [36], and Replica [161] for cross-modality scene query tasks. To NeRF these scenes, we trained 4000 steps for Objectron

Table 4.7. Depth estimation with frozen features. We report performance when training a linear classifier on top of one (lin. 1) or four (lin. 4) transformer layers. We report the RMSE metric on the 3 datasets. Lower is better.

	NYUd		NYUd → SUN RGBD	
Method	lin. 1	lin. 4	lin. 1	lin. 4
OpenCLIP [71]	0.541	0.510	0.537	0.476
MAE [54]	0.517	0.483	0.545	0.523
DINO [18]	0.555	0.539	0.553	0.541
iBOT [231]	0.417	0.387	0.447	0.435
DINOv2 [123]	**0.344**	**0.298**	0.402	**0.362**
Ours	0.361	0.322	**0.389**	0.367

Table 4.8. Stereo depth estimation on the DTU dataset. Backbone methods could benefit a lot from our feature initialization.

Model	Overall Err.
MVSNet [205]	0.462
PointMVS [28]	0.366
DINOv2 [123] w/ MVSNet	0.389
DINOv2 [123] w/ PointMVS	0.365
ConDense w/ MVSNet	0.341
ConDense w/ PointMVS	**0.320**

and 8000 steps for both ScanNet and Replica on each scan. We use ground-truth bounding boxes included in these datasets. For "Ren5" baselines, we render images in the same resolution as in their corresponding datasets and the 5 views are randomly sampled from the original camera trajectories. So these 2D-Native methods are taking advantage of that queries and indices are drawn from the same trajectory, where in real-world cases this is not possible– since the original image sequences and trajectories are typically not accessible in 3D models. For ULIP-2 [200], we use its global scene embedding to perform the top-1 matching between the queries and indices. For our methods, we use 32 key points for each 2D image and 32, 64, and 64 key points for 3D input from Objectron, ScanNet, and Replica respectively. For our 2D-3D key point matching, we use the threshold 0.75

and select the 3D scene with the most number of successful matches as the query result.

3D Scene Retrieval with a Single Image (2D-3D). In this task, we retrieve a scene from a repository with a single view. To compare with 2D-only methods, we first render 5 views (Ren5) from a scene and compute the cosine similarity of the query image and rendered views, and then use the winner-take-all scheme to identify the scene.

For Objectron [170], we sampled 1000 scenes spreading over 9 object categories. For each scene, we sample one image as the test query. We use top-1 to match between queries and keys and calculate the retrieval accuracy. For ScanNet [36], we use all 1513 scans. We sample one frame as a test query per 100 frames and at least one frame for every scene regardless of its length.

In Table 4.9, it can be seen ConDense 2D is a strong baseline not only outperforming all the other 2D methods but also performing better than ULIP-2. In 2D-3D methods, both global (using globally averaged feature) and KP (key point matching with RANSAC) variants of ConDense do better than the other 2D-3D method.

Table 4.9. 3D scene retrieval and 3D scene duplicate detection results on multiple datasets with frozen features. The upper includes 2D solutions where scenes are represented by 5 random views (Ren5), and the lower includes 2D-3D native methods.

	3D Retrieval (Acc)		3D Dup. Det. (AP^*_{75})	
Model	Objectron	ScanNet	ScanNet	Replica
OpenCLIP (Ren5) [71]	90.3	49.8	51.0	52.7
DINOv2 (Ren5) [123]	88.1	43.1	41.3	43.3
Unicom (Ren5) [4]	92.9	52.5	54.3	57.0
ConDense 2D (Ren5)	**94.6**	**53.3**	**58.7**	**59.0**
ULIP-2 [200]	89.7	61.7	63.0	66.6
ConDense-Global	91.6	70.1	65.3	66.9
ConDense-KP	**92.9**	**78.4**	**70.7**	**72.0**

3D Scene Duplicate Detection (3D - 3D). We further test the matching capabilities of ConDense at the scene level by proposing a new task of detecting duplicate scenes in a large NeRF repository. Our method is generalizable to both NeRF and Point-Cloud inputs, and we run our experiments on ScanNet and Replica [36, 161]. For both datasets, we sample 300 scene pairs where half of them are NeRFs from the same scene (duplicates) and half of them not. For ScanNet, the duplicates are created from different scans of the same scenes. (*e.g.* scene #0 has 3 different scans.) For Replica, the duplicates are created from the overlapping scans of the same scenes, where at least 50% of trajectory overlappings are ensured. In this way, we comprehensively test the 3D matching capability of models on either full scans or adjacent partial scans. We use 0.75 as the threshold when determining if two embeddings belong to the same scene as a way to detect duplicated scenes. AP^{*}_{75} is calculated as the classification accuracy of duplicate detection when the threshold is 0.75.

Results are presented in Table 4.9. Here, Ren5 methods are similarly defined as in the previous 3D scene retrieval, where scenes are rendered into images and the image embeddings are used. We use $\theta = 0.75$ as the threshold to determine if two embeddings belong to the same scene. Here we find the remarkable effectiveness of our key points. There is a large gap between 2D-3D methods for this task, showing the need to tackle this task in 3D feature space. While ConDense-Global performs similarly to ULIP-2, ConDense-KP is significantly better for scene-to-scene matching.

4.7 Model Analysis and Discussions

4.7.1 Ablation Study

Table 4.10. Ablation study on removing individual components in our pre-training pipeline, evaluated on both 3D classification (Overall Acc on ScanObjectNN, before slash) and 2D classification (Acc on ImageNet-1k, after slash).

Full Model	Freeze 2D	No $\mathcal{L}_{\text{p}}$	No $\mathcal{L}_{\texttt{fid}}$	10% Data
94.1 / **89.6**	91.3 / 86.5	90.5 / 87.1	89.7 / 79.9	88.7 / 79.1

We perform experiments to verify the effectiveness of our design. The ablation

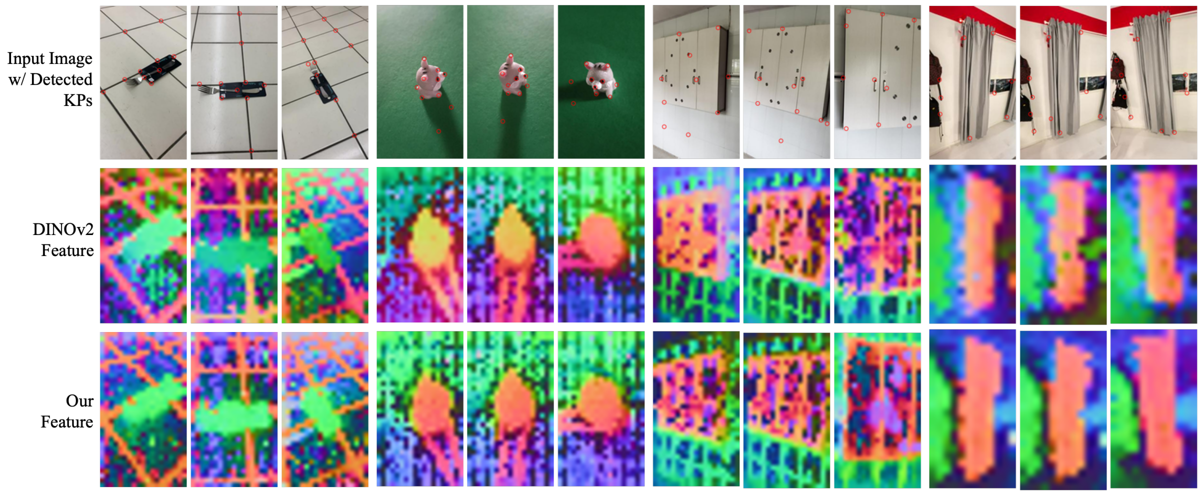

Figure 4.4. Visualization of our 2D dense feature reveals its superiority over the Original DINOv2 feature in terms of consistency across multi-view images. Additionally, we present visualizations of sparse feature locations identified by our key point detector.

study results are presented in Table 4.10. **Freeze the 2D encoder?** When we freeze the 2D encoder, as is done in other methods [130, 199], we observed a worse performance in performance for both 2D and 3D tasks. We can also see that the features from our backbone are more 3D consistent and contains more detail. Please check the visualization in our supplementary materials. **Sparse feature helps?** The sparse feature module is an integral component of our framework, not only enabling novel capabilities in 2D-3D retrieval but also serving as a strong self-supervision signal to enhance performance on individual 2D and 3D tasks. **2D fidelity helps?** The 2D fidelity loss helps prevent the 2D features from collapsing to a trivial solution or overfitting the biased data distribution. The exclusion of the 2D fidelity module has a detrimental impact on the quality of both 2D and 3D tasks, as evidenced by our experiments. Part of this loss is due to multi-view datasets being still considerably smaller than 2D image datasets, and featuring mostly man-made objects and indoor scenes. The limited size and biased distribution can cause the features to deviate significantly, leading to worse results.

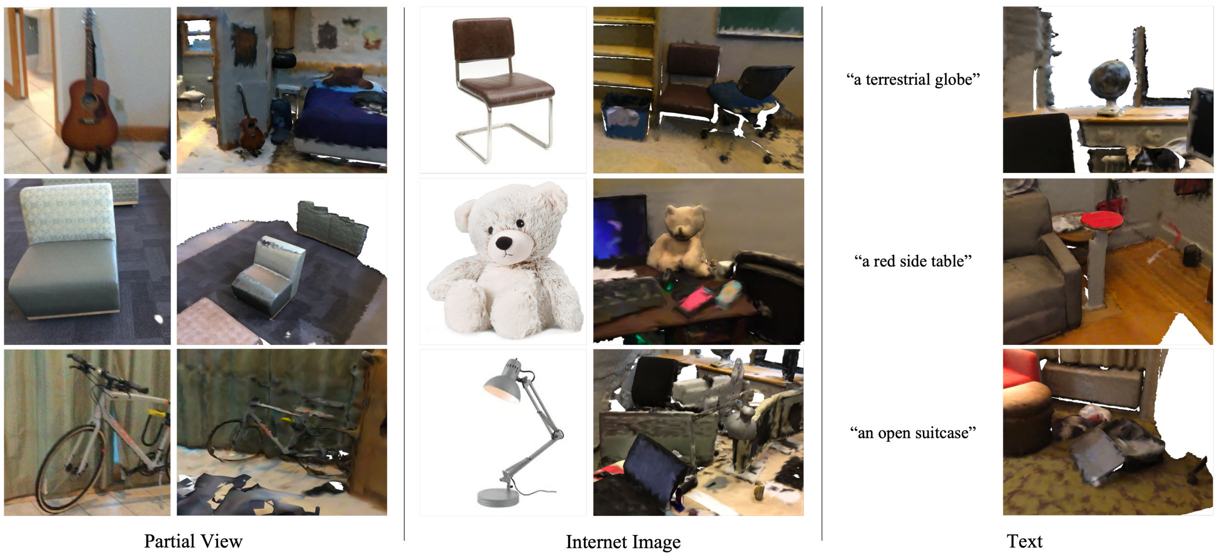

Figure 4.5. Visualization of using different types of input to query the target scene repository (ScanNet). Within each pair are query inputs (left) and top-1 query results (right).

4.7.2 Feature Visualization

For Figure 4.4, we ran PCA on the generated feature maps, both DINOv2 (2nd row) and ours (3rd row), of each scene and selected top-3 components to visualize them as red, green, and blue, respectively. We also visualize the top 10 confident (in terms of predicted probabilities) key points with positive 3D matches in this figure (1st row).

4.7.3 Cross-Modality Queries

We show in Figure 4.5 the visualization of using different types of input to query the target 3D scene repository (ScanNet). The unique 2D-3D co-embedded embedding space with sparse key point design enables ConDense to effectively query objects in repositories of large scenes. Here, ConDense is not only able to query 3D scenes with partial views from the dataset but also able to find objects in these scenes that match the appearance from unseen internet images. In addition, by changing our backbone to a multi-scale CLIP [140] feature as in [80], we further acquire the ability to query 3D scenes with natural languages inputs, and thus build a language-image-3D co-embedded feature space with sparse key points. With this modified model, we can query scene repositories with text inputs as in Figure 4.5 third row.

Table 4.11. 3D Classification (Acc) on MVImgNet, Co3D, and ShapeNet. Our method is even more useful when 3D training data are scarce (ShapeNet 1%).

Method	MVImgNet	Co3D	ShapeNet (1%)	ShapeNet (10%)	ShapeNet (Full)
From Scratch	73.5	81.1	61.2	76.9	84.9
PointContrast [194]	76.9	84.9	65.8	78.9	86.6
Point-MAE [126]	79.1	86.2	72.3	81.1	89.2
ULIP-2 [200]	82.9	86.1	74.3	81.9	89.3
Ours (Freeze 2D)	87.3	88.8	80.1	83.9	90.1
Ours	**91.3**	**93.8**	**81.4**	**85.3**	**91.9**

Table 4.12. 3D Segmentation (mIOU) on ScanNet, SemanticKitti, and S3DIS. Our method is even more useful when 3D training data are scarce (ScanNet 1%).

Method	ScanNet (1%)	ScanNet (10%)	ScanNet (100%)	SemanticKitti	S3DIS
PointNet [135]	42.2	62.1	72.2	19.6	47.6
Mix3D [119]	39.4	69.9	73.6	65.4	63.5
PointContrast [194]	52.9	70.4	74.1	71.7	75.2
Swin3D [204]	54.8	65.2	77.5	74.7	79.8
Ours (Freeze 2D)	65.6	70.0	78.1	74.6	80.6
Ours	**67.3**	**72.3**	**79.8**	**75.1**	**80.7**

4.7.4 Effect of Data Amount

We tested our 3D capabilities on more datasets and also with varying amounts of 3D training data. The results are presented in Table 4.11 (classification) and Table 4.12 (segmentation). We see consistent improvement over all baselines from these results. Also, it can be seen that our method is even more useful when 3D training data is very scarce (See ShapeNet 1% and ScanNet 1%).

Table 4.13. Comparing different backbones.

	PointBERT	PointNeXt	MinkowskiNet
#Parameters	32.3M	41.6M	41.3M
3D Cls / 3D Seg	87.2 / -	92.1 / 77.0	**93.2** / **79.1**

4.7.5 Backbone Architecture

The MinkowskiNet (MNet) is the established state-of-the-art for dense 3D tasks and has been adopted by most 3D dense task models. So we select MinkowskiNet as our backbone. Here we had an experiment comparing different backbones – results are reported in Table 4.13 on ScanObjectNN (3D Cls) and ScanNet (3D Seg).

We see better performance, especially on 3D Seg, with MNet. The reasons behind this may include **(1)** an overfitting tendency of the transformer-based model PointBERT, which aligns with OpenScene's finding (their Sec. 6.4); **(2)** our better generalizability from training to test domains, where point distributions are different. We will include a more detailed discussion on the performance and scalability of different backbones once time permits.

Table 4.14. Ablation study on NeRF quality in the pre-training dataset.

Training NeRF Type	Mip-NeRF (2k steps)	Mip-NeRF (4k steps)
Quality (PSNR/SSIM)	28.54 / 0.899	30.23 / 0.939
2D Cls / 3D Cls	86.2 / 88.9	**87.7** / **93.2**

4.7.6 Effect of NeRF Quality

We observed differences in performance when using trained NeRFs of different quality. The numbers are reported on ImageNet (2D Cls) and ScanObjectNN (3D Cls) on a lighter version of the final model reported in the main paper.

Results are presented in Table 4.14. We find that pre-training with data of lower quality will have an impact on performance, especially for 3D tasks. We take measures such as using different iteration numbers to ensure convergence, and filtering out low-quality frames as mentioned previously.

4.8 Conclusions and Limitations

In this work, we have presented ConDense, a framework for 3D pre-training that adeptly harmonizes 2D and 3D feature extraction using pre-trained 2D networks and multi-view datasets, in both the dense and the sparse feature regimes. Our approach not only provides a pre-trained 3D network acing in multiple tasks but also enhances the quality of 2D feature representation and establishes a unified embedding space for multi-modal data interaction. Extensive experiments demonstrate the superiority of ConDense over existing 3D pre-training methods in tasks like 3D classification and 3D segmentation and in new applications such as 2D image queries of 3D NeRF scenes. ConDense marks an advance in 3D computer vision, reducing the reliance on scarce 3D data and enabling more efficient and multi-modality queries on 3D scenes.

Nonetheless, the pipeline comes with several limitations including the relatively high cost of processing multi-view data. Exploring more efficient ways to exploit multi-view data as well as ready-to-use 3D data could be a useful future direction. Furthermore, combining techniques from contrastive learning methods [194, 56] and efficient fine-tuning methods (*e.g.* LoRA [67]) could improve the overall robustness and efficiency of the pipeline. We believe that ConDense opens up exciting avenues for model pre-training and 2D/3D feature backbones, and defer these to future research in this direction.

Acknowledgements

Chapter 4, in full, is a reprint of the material published in the 2024 European Conference on Computer Vision (ECCV): "ConDense: Consistent 2D/3D Pre-training for Dense and Sparse Features from Multi-View Images" (Xiaoshuai Zhang, Zhicheng Wang, Howard Zhou, Soham Ghosh, Danushen Gnanapragasam, Varun Jampani, Hao Su, and Leonidas Guibas). The dissertation author was the primary investigator and author of this paper. Our thanks go to Hao-Ning Wu and Bhav Ashok for their support in building large-scale pose estimation and NeRF pipeline for dataset pre-processing.

Chapter 5

Fast 3D Scene Generation by Lifting Panorama Images from 2D Diffusion Models

5.1 Introduction

The creation of immersive 3D environments represents a frontier in computer vision and graphics, promising to revolutionize content creation for virtual reality, gaming, and architectural visualization. While recent years have seen remarkable progress in image synthesis, particularly with the advent of diffusion models, extending these capabilities to full 3D scene generation introduces a host of new challenges. These challenges stem from the need to ensure geometric consistency, scene completeness, and the ability to render novel views that were not explicitly generated.

In this dissertation, we propose and analyze a framework for text-to-3D and image-to-3D scene generation that bridges the gap between the rich priors of 2D diffusion models and the complexities of 3D scene representation. Our approach leverages the strengths of existing 2D generative models to bootstrap a robust 3D reconstruction process, effectively decomposing the problem into manageable stages that can be optimized independently.

The core of our method lies in a two-stage pipeline: First, we harness state-of-the-art 2D diffusion models, specifically panorama generation models, to produce photorealistic panorama images. These models, which can be conditioned on text descriptions or single image inputs, generate panorama images with a comprehensive 360-degree representation of the scene. In the second stage, we employ a learned 3D reconstruction model that integrates information across these generated images to infer the underlying 3D geometry and appearance. This reconstruction process is designed to complete any missing regions, resulting in a coherent 3D scene that can be freely navigated.

Our two-stage pipeline offers several key advantages:

Flexibility in Input Modalities. Our framework supports both text-to-3D and image-to-3D generation, providing flexibility for various use cases and user preferences. This versatility makes our approach applicable to a wide range of scenarios, from purely imaginative creations to extensions of existing environments.

Complete, Navigable 3D Scenes. Unlike methods that focus on generating single objects or constrained viewpoints, our approach produces complete, navigable 3D scenes. This enables users to explore the generated environments from any angle, enhancing the immersive experience.

Scalability and Efficiency. By decomposing the problem into 2D generation followed by 3D reconstruction, we can leverage highly optimized 2D diffusion models and focus our 3D learning efforts on the reconstruction task. This leads to a more scalable and efficient pipeline and an easier optimization process.

We demonstrate through experiments that our approach generates diverse, high-quality 3D scenes from text descriptions or single images. By enabling the rapid creation of immersive 3D environments from simple text descriptions, we open up new possibilities for creative expression, prototyping, and virtual world-building. This technology has the potential to democratize 3D content creation, making it accessible to a broader audience of artists, designers, and developers who may lack traditional 3D modeling expertise.

In summary, this chapter introduces a new methodology for text-to-3D and image-to-3D scene generation that leverages the strengths of 2D diffusion models to produce high-quality, navigable 3D environments. By bridging the gap between 2D and 3D generation, we pave the way for more advanced and accessible 3D content creation tools, bringing us closer to the vision of effortlessly translating imagination into immersive virtual experiences.

5.2 Related Work

Our proposed 3D scene generation from the panorama prior pipeline is built upon the recent advancement of 2D panorama generative and 3D scene implicit representations. We now review related work on these techniques as well as similar approaches that integrate 2D generative model priors into object-level 3D generation tasks.

Panorama Generative Model. Omnidirectional 360-degree panorama images are subjected to nonlinear perspective distortion of equirectangular projection, resulting in a shift of data distribution from regular 2D images used to train large-scale image generative models. Among existing methods, one line of works [176, 2, 179, 122, 188, 38, 157, 53] tries to solve the problem of panorama image generation as an outpainting problem and focuses on generating a 360-degree panorama from a partial image input. To tackle the distribution shift while ensuring visual authenticity, various attempts have been made. OmniDreamer [2] utilized transformer-based networks to perform diverse completions and samplings while BIPS [122] integrates depth-aided adversarial supervision into their training process. PanoDiffusion [189] adopts progressive camera rotations during each diffusion denoising step, leading to panorama wrap-around consistency improvements. Alternatively, following the success of advanced generative models, another line of works explores to synthesize high-fidelity panorama images, from textual input [64, 158, 188, 43, 181, 201, 210, 177]. Such attempts integrate rich prior information from pre-trained 2D text-to-image generation models, further broadening their generalization capability to zero-shot scenarios. Test2Light [30] introduce VQGAN [42] structure to synthesize panorama images from text inputs. AOGNet [109] performs 360-degree outpainting through an autoregressive process but suffers from inefficiency in the sampling process. MVDiffusion [167] generates multi-view consistent images indirectly which can be stitched together into panorama images. More recently, PanFusion [215] proposes a dual network, considering both global panorama views and local perspective views, ensuring both global consistency and local quality.

3D Reconstruction with Implicit Representations. Neural rendering techniques have recently gained significant attention for their ability to represent high-quality 3D scenes for realistic rendering, novel view synthesis, and extended applications. Most recent methods are based on NeRF [11] which represents scenes as volumetric neural radiance fields [22, 45, 117] and is optimized by per-scene rendering supervision. Given the expressibility of NeRF, the NeRF-based pipeline suffers from high computation overhead, as rendering usually requires dozens of queries through large MLP of the neural field per ray. 3D Gaussian Splatting (3DGS) [78] solves this problem by representing the radiance field using 3DGS that can efficiently be rendered via rasterization. However, all these methods require dense input images and time-consuming per-scene optimization to achieve high-quality reconstruction. More recently, transformer-based large reconstruction models (LRMs) emerged as a general framework [65, 197, 182, 216, 184, 198] for reconstructing an implicit 3D representation (NeRF or 3DGS) with only a feed-forward pass. These methods learn from a large-scale dataset and capture rich priors useful for 3D reconstruction, and achieve high-quality 3D reconstruction with only sparse view inputs. LRM [65] adopts a highly scalable transformer-based architecture to directly predict a neural radiance field from the input images. CRM [182] generates six canonical view images from a single image input, and then feeds these images into a convolutional U-Net to create a high-resolution triplane neural field. These LRM-derived methods mostly focus on object-level reconstruction. For scene-level reconstruction, a more common line of research is to leverage per-pixel (per-patch) feature and backproject each pixel into 3D space [21, 164, 29, 218]. PixelSplat [21] leverages epipolar line-based sampling to overcome local minima. GS-LRM [218] takes advantage of plucker ray coding [133] and directly processes image patches as a sequence of variable length. Flash3D [164] integrates a monocular depth estimation model as prior and extends it to a full 3D shape and appearance reconstruction. MVSplat [29] builds a cost volume representation via plane sweeping which provides valuable geometry cues to the estimation of depth. This line of work is the most related to our proposed method, while we adopt a few novel techniques to take care of unique challenges in full 3D scene generation.

3D Object Generation from 2D Prior. The advancements in generative techniques [81, 62] enable large-scale pretrained 2D generative models to synthesize high-fidelity, open-vocabulary, photorealistic 2D visual contents [41, 63]. There is a host of new challenges to extending these capabilities into 3D content generation. Vast explorations have been made into utilizing 2D diffusion models in object-level 3D generation. DreamFusion [134] introduces the Score Distillation Sampling (SDS) method and a loss based on probability density distillation that enables the use of a 2D diffusion model as a prior for optimization of the objective neural field parameters. Later works [181, 155, 70, 104] improve SDS sampling in terms of training speed, cross-view consistency, and texture fidelity. Notably, MVDream [155] trains a multi-view diffusion to generate multi-view images simultaneously, improving the 3D awareness of 2D. Later works [103, 153, 101, 99] take advantage of consistent multi-view generation models to further improve the quality and speed of 3D object generation. However, these methods do not trivially extend to 3D scene generation due to inherent challenges including lack of large-scale training data and limited representation capabilities.

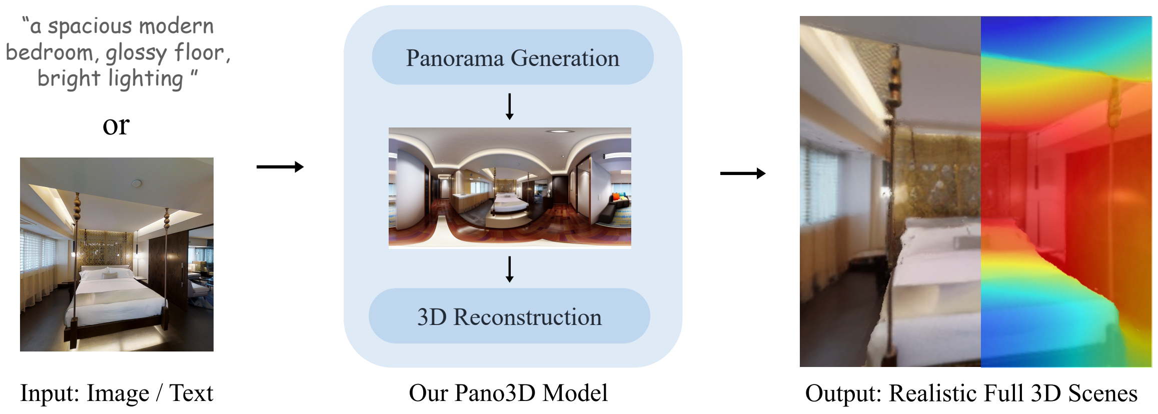
"a spacious modern bedroom, glossy floor, bright lighting"
or
Input: Image / Text
Panorama Generation
3D Reconstruction
Our Pano3D Model
Output: Realistic Full 3D Scenes

Figure 5.1. Given a text or single image input, the pipeline first employs a 2D panorama generation model to generate photorealistic panorama images, and then perform 3D reconstruction to create complete and navigable 3D scenes. Pano3D efficiently creates 3D scenes from only feed-forward passes.

5.3 Method

Our pipeline (Figure 5.1) integrates off-the-shelf panorama generative models to synthesize 360-degree panorama images $I \in \mathbb{R}^{H\times W\times 3}$ conditioned on given inputs (text or image). Then the panorama images are processed by our novel 3D Gaussian-based reconstruction model to produce full 3D scenes with only fast feed-forward passes.

5.3.1 Panorama Image Generation

For text-to-panorama generation, we adopt the FLUX.1 [dev] model [88], since it is the current state-of-the-art for image generation. The original FLUX.1 [dev] is distilled from FLUX.1 [pro], a giant model, which is trained on a diverse set of natural, art, and artificial images. We finetune FLUX.1 [dev] with LoRA on our curated panorama image dataset for 2000 steps. The dataset contains 72 samples selected from public-domain internet images, and we captioned the images manually.

For image-to-panorama generation, we adopt the Diffusion360 model [44]. The model takes a perspective image of arbitrary shape and a corresponding mask on a panorama image as input, and outpaint the entire panorama image based on the content.

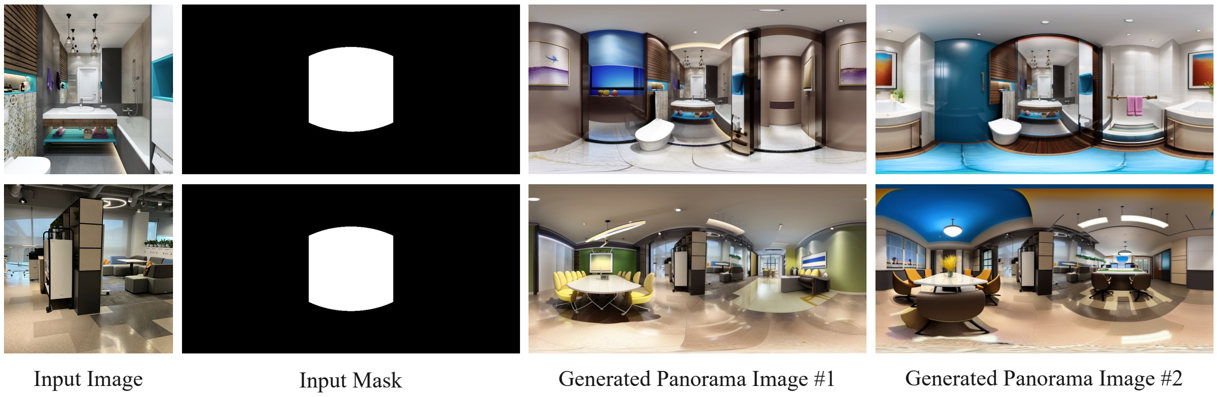

Figure 5.2. Generation examples with single image inputs. The model could generate a diverse set of panorama images with different inputs or different diffusion runs.

We use a standard center mask and square image inputs throughout the paper to simplify the settings. Examples of these images and masks are shown in Figure 5.2.

5.3.2 Navigable 3D Scene from a Single Panorama Image

As shown in Figure 5.3, given the generated 2D panorama image I, we then use a simple transformer-based network to convert the image into a complete and navigable 3D Gaussian splat-based scene.

Transformer-based Reconstruction Model

We use a transformer-based model to regress 3D Gaussian primitive parameters corresponding to each pixel in the image. Given a panorama image I, we first incorporate the spatial information into the pixel feature by concatenating Plücker ray coordinates [133] following previous works [196, 25]. Specifically, for each pixel i, j we compute the $r_{i,j} = (o \times d_{i,j}, d_{i,j})$ in world coordinate; here o is the origin of the camera, and $d_{i,j}$ is the normalized direction vector from the origin to the pixel, calculated by reverse equirectangular projection. The 6D Plücker ray coordinates are concatenated with the image RGB colors channel-wise, forming 9-channel inputs to the network.

Following ViT [41], we patchify input pose-conditioned panorama images I into

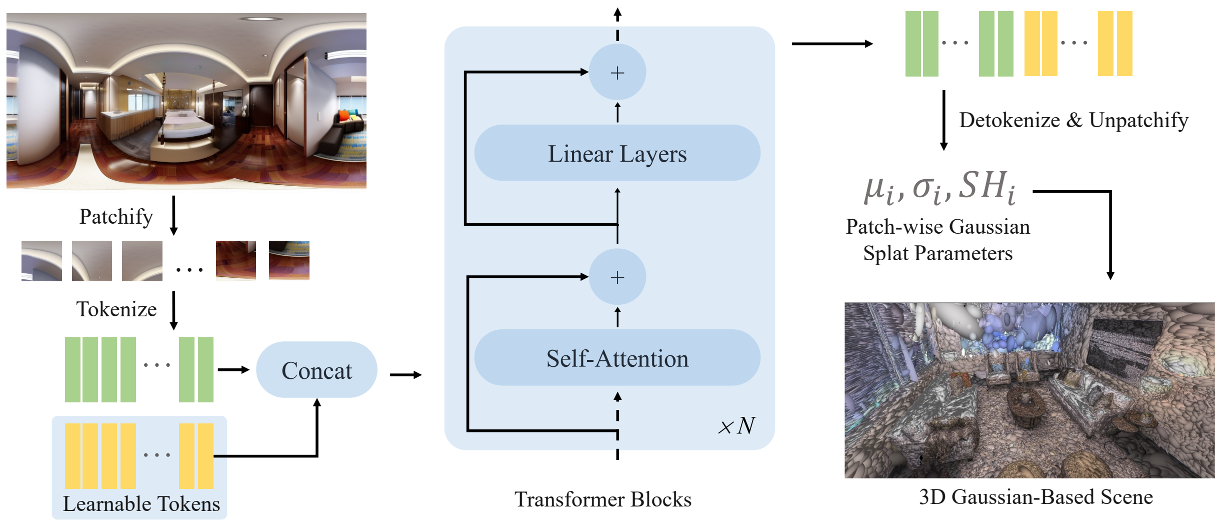

Figure 5.3. The input panorama image is pachified and processed into a sequence of tokens with flatten and linear operations. The transformer network maps this sequence of tokens into patch-wise Gaussian splat parameters, and constructs the final 3D Gaussian based scene.

non-overlapping 2D tokens $\{T_{i,j} \in \mathbb{R}^{h \times w \times 9}\}$ indexed by x, y, where $h = \frac{H}{p}, w = \frac{W}{p}$, and p is the patch size (set to 8 for all experiments). These tokens are flattened into a 1D sequence of width $9 \cdot p^2$ and length $h \cdot w$ before being transformed by a set of linear layers into dimension d to align with the following transformer layers. No positional encoding is used in this network since absolution location information is already encoded in the Plücker ray coordinates. If more than one image is passed to the network, we simply concatenate the tokens as a longer sequence. This further improves the overall flexibility of the pipeline, making it easier to train and use under different settings.

The pose-conditioned tokens are fed into layers of residual connected [58] transformer blocks [171]. Each consists of a self-attention layer and a set of linear layers as implemented in ViT [41]. We then take the output tokens and decode them into Gaussian primitive parameters with a set of linear layers, each token produces p^2 sets of Gaussian parameters, representing p^2 pixels within the original patch. This step generates $h \times w$ Gaussian primitives in total.

We follow [78] and parameterize the 3D Gaussian primitives by 3-channel RGB, 3-channel scale, 3-channel location, 4-channel rotation quaternion, and 1-channel opacity, and 1-channel distance (to the corresponding pixel center).

Learnable Gaussian Tokens for Scene Completion

The formulation above could generate 3D Gaussian primitives and cover most portion of the scene. However, such a Gaussian placement strategy is not capable of covering the occluded parts of the scene, leaving the scene incomplete.

To generate a complete 3D scene, the ability to generate free-float 3D Gaussian primitives is required. This is solved by introducing learnable tokens into the pipeline. Given the tokenized sequence, we concatenate a set of learnable tokens at the end of the sequence. Then at the detokenize and unpatchify steps, each of these tokens decodes into p^2 3D Gaussian primitives, but with 3-channel location instead of 1-channel distance. This enables the network to reason where to add 3D Gaussian primitives and complete the scene as much as possible, leaving much fewer holes.

Loss Terms and Training Scheme

During training, we render panorama images from the generated 3D Gaussian primitives at 10 supervision views. The views are randomly sampled within the scene beforehand. Following [218], we use both perceptual loss and mean squared error (MSE) loss terms during training. We also followed their practice of using VGG-19 [156] based perceptual loss.

We adopted the multi-stage training scheme to reach convergence faster and make the most of the limited 3D scene data. In the first stage, we train the network with perspective images from the RealEstate10k [234] dataset. We use 2-6 input images in this stage. Both input images and supervision images are randomly sampled on the same video sequence, encouraging the network to learn extrapolation and generation of unseen

Gaussian primitives in the input images. For the second and third stages, we switch to our own 3D synthetic dataset, which contains around 10k 3D indoor scenes. In the second stage, we use 6 cube map images as input to the network and supervise with randomly sampled perspective views within the scene. This stage adapts the network to single-viewpoint data. In the final stage, we use a single panorama image as input and supervise with randomly sampled perspective views within the scene. This prepares the network for our actual test settings.

The training takes around 60 hours with 32 NVIDIA H100 GPUs, in which the three stages take around 30, 15, and 15 hours respectively. The network acts fast during evaluation, taking less than 1s with 1024×512 panorama images on a single NVIDIA H100 GPU.

5.4 Experiments

In this section, we perform various experiments to evaluate the performance of the proposed Pano3D framework, both quantitatively and qualitatively.

5.4.1 3D Scene Generation

Text-Conditioned Generation

Our results for text-conditioned generation are presented in Figure 5.4. For the same input text prompt, our model is capable of sampling different 3D scenes, as shown in the first and second columns. Our model generates smooth geometry and realistic appearance, leading to appealing 3D scenes. This setting is particularly useful for VR devices since users could be able to enter and explore new 3D scenes given solely their text prompts.

Table 5.1. Quantitative analysis of novel view synthesis quality. The experiments are performed on the Matterport3D dataset [19].

Method	PSNR ↑	SSIM ↑	LPIPS ↓
NeRF [11]	18.32	0.745	0.341
TensoRF [22]	18.90	0.778	0.333
OmniNeRF [66]	33.25	0.954	0.110
Pano3D (Ours)	**33.98**	**0.961**	**0.098**

Table 5.2. Quantitative analysis of geometry accuracy of generated 3D scenes. The experiments are performed on the Matterport3D dataset [19].

Method	Abs Rel ↓	Sq. Rel ↓	RMSE ↓
OmniFusion [92]	0.1007	0.0969	0.4435
HRDFuse [1]	0.0967	0.0936	0.4433
Liu *et al.*, 2024 [97]	0.0941	0.0723	0.4396
Pano3D (Ours)	**0.0889**	**0.0613**	**0.4126**

Image-Conditioned Generation

Our results for image-conditioned generation are presented in Figure 5.5. Our model is capable of generating different types of scenes ranging from bedrooms to bathrooms, given the image condition. Our model not only generates 3D scenes with appealing appearance and geometry details but also well follows the input condition, making it suitable for a wide range of downstream tasks such as interior designing and remodeling.

For quantitative evaluation of image-conditioned generation, we followed the settings of OmniNeRF [66], where they train the models on a single panorama image and evaluate on a set of neighboring views. The results on the Matterport3D dataset are shown in Table 5.1. Our model generates the scene that best matches the original inputs.

5.4.2 Geometry Accuracy

To evaluate the geometry quality of the generated scenes, we test on the Matterport3D dataset by generating 3D scenes from Matterport3D panorama images and comparing rendered depth maps with the dataset ground truth. We also include several recent state-of-the-art methods for panoramic depth estimation [92, 1, 97]. The results

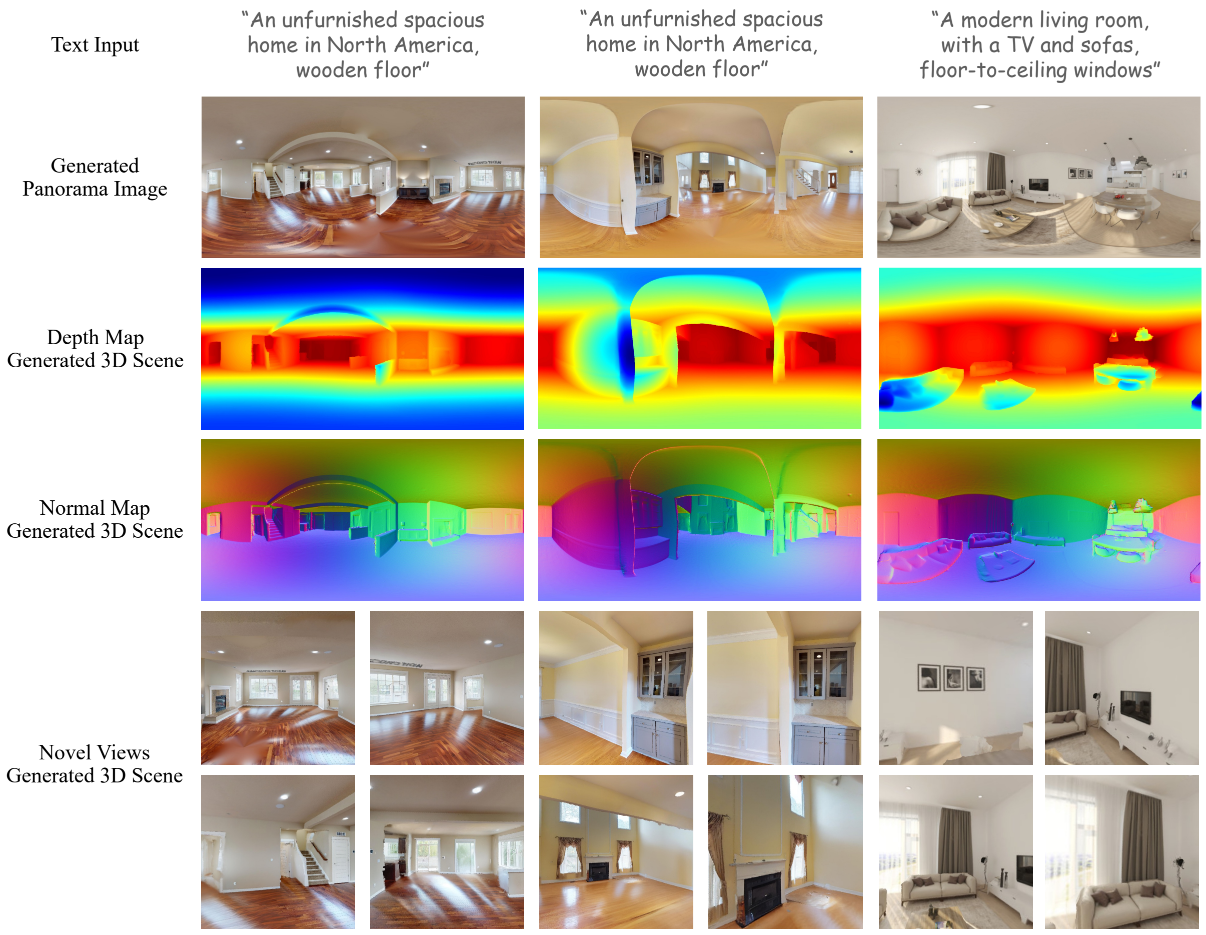
Text Input
"An unfurnished spacious home in North America, wooden floor"
"An unfurnished spacious home in North America, wooden floor"
"A modern living room, with a TV and sofas, floor-to-ceiling windows"
Generated Panorama Image
Depth Map Generated 3D Scene
Normal Map Generated 3D Scene
Novel Views Generated 3D Scene

Figure 5.4. 3D scene generation examples with text inputs. The model could generate 3D scenes with consistent geometry and good novel view quality.

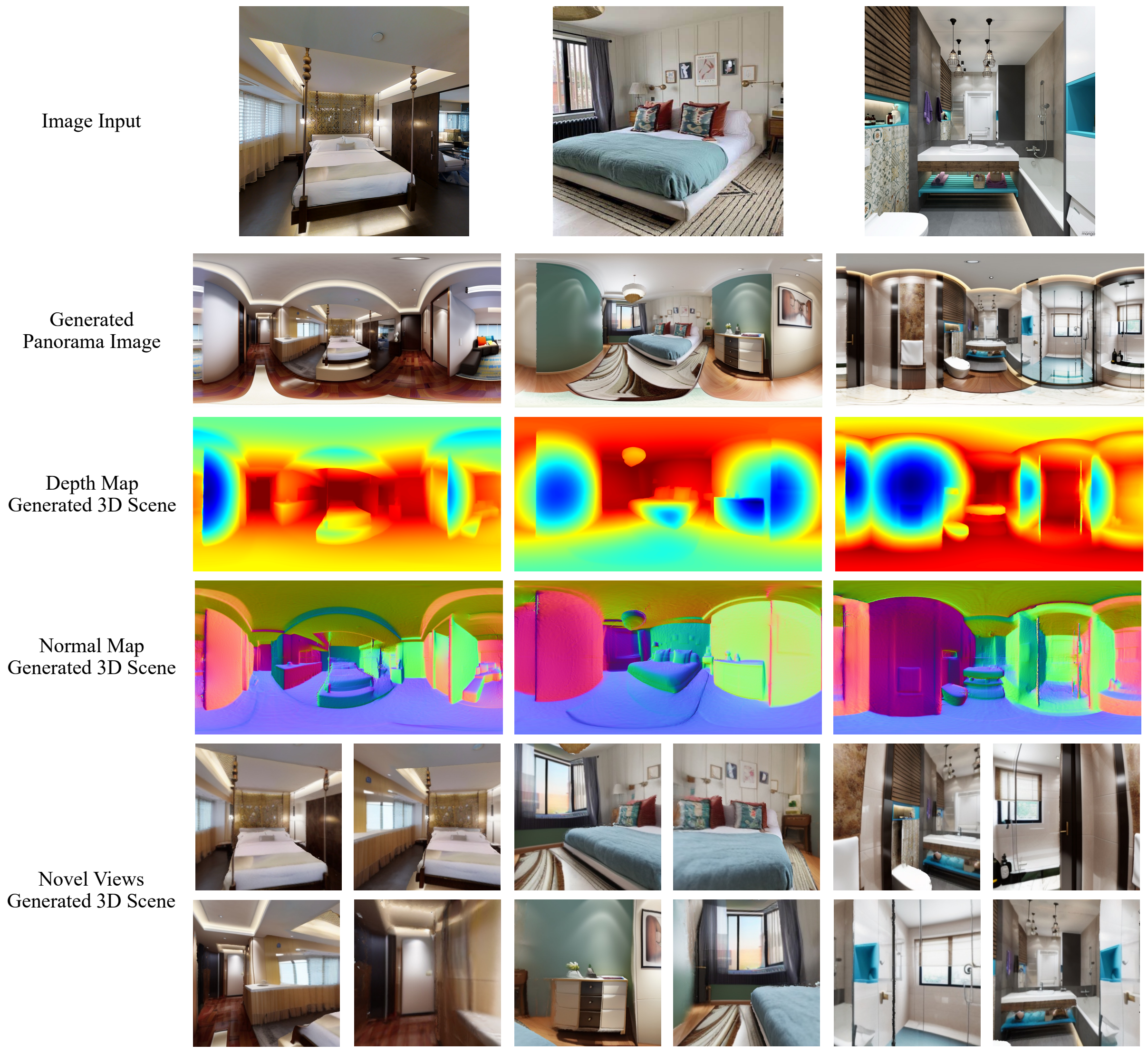
Image Input
Generated Panorama Image
Depth Map Generated 3D Scene
Normal Map Generated 3D Scene
Novel Views Generated 3D Scene

Figure 5.5. 3D scene generation examples with single image inputs. The model could generate 3D scenes with consistent geometry and good novel view quality.

are presented in Table 5.2. We find that our model outperforms all other models, even though no explicit geometry supervision is given in the training stage, suggesting our strong capability of geometry reasoning.

5.5 Conclusion and Limitations

This chapter presents a framework for 3D indoor scene generation that integrates 2D panorama image diffusion models with 3D scene reconstruction methods, addressing key challenges in creating immersive, navigable virtual environments. By leveraging the strengths of cutting-edge 2D generative techniques, our two-stage pipeline provides a flexible, efficient approach to transforming text descriptions and single images into comprehensive 3D scenes. The proposed transformer-based reconstruction network and panorama-oriented training scheme help improve the completeness and visual quality of the generated scenes. This approach points to a clear direction for making realistic 3D data more accessible to a diverse range of users and applications.

The method comes with several shortcomings. First, it is not possible to generate large-scale outdoor scenes with this pipeline, due to the inherent difficulty of modeling distant 3D Gaussian primitives. Secondly, the method is prone to distance ambiguity due to its single view nature. Because of this, the method would sometimes generate curved walls and ceilings, which is rarely seen in real world. The method also relies heavily on the consistency and correctness of the 2D generation model. Any failure in the first step generation (*e.g.* prompt not matched, or geometrically incorrect image) is not fixable and the final result would also fail. An interesting future direction is to incorporate more explicit 3D priors to solve these issues.

Bibliography

[1] Hao Ai, Zidong Cao, Yan-Pei Cao, Ying Shan, and Lin Wang. Hrdfuse: Monocular 360deg depth estimation by collaboratively learning holistic-with-regional depth distributions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13273–13282, 2023.

[2] Naofumi Akimoto, Yuhi Matsuo, and Yoshimitsu Aoki. Diverse plausible 360-degree image outpainting for efficient 3dcg background creation. In *CVPR*, pages 11441–11450, 2022.

[3] Kara-Ali Aliev, Artem Sevastopolsky, Maria Kolos, Dmitry Ulyanov, and Victor Lempitsky. Neural point-based graphics. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXII 16*, pages 696–712. Springer, 2020.

[4] Xiang An, Jiankang Deng, Kaicheng Yang, Jaiwei Li, Ziyong Feng, Jia Guo, Jing Yang, and Tongliang Liu. Unicom: Universal and compact representation learning for image retrieval. *arXiv preprint arXiv:2304.05884*, 2023.

[5] Iro Armeni, Zhi-Yang He, JunYoung Gwak, Amir R Zamir, Martin Fischer, Jitendra Malik, and Silvio Savarese. 3d scene graph: A structure for unified semantics, 3d space, and camera. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5664–5673, 2019.

[6] Iro Armeni, Ozan Sener, Amir R. Zamir, Helen Jiang, Ioannis Brilakis, Martin Fischer, and Silvio Savarese. 3d semantic parsing of large-scale indoor spaces. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1534–1543, 2016.

[7] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495, 2017.

[8] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5855–5864, 2021.

[9] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. *CVPR*, 2022.

[10] Sai Bi, Nima Khademi Kalantari, and Ravi Ramamoorthi. Patch-based optimization for image-based texture mapping. *ACM Transaction on Graphics*, 36(4):106–1, 2017.

[11] Sai Bi, Zexiang Xu, Pratul Srinivasan, Ben Mildenhall, Kalyan Sunkavalli, Miloš Hašan, Yannick Hold-Geoffroy, David Kriegman, and Ravi Ramamoorthi. Neural reflectance fields for appearance acquisition. *arXiv preprint arXiv:2008.03824*, 2020.

[12] Sai Bi, Zexiang Xu, Kalyan Sunkavalli, Miloš Hašan, Yannick Hold-Geoffroy, David Kriegman, and Ravi Ramamoorthi. Deep reflectance volumes: Relightable reconstructions from multi-view photometric images. In *Proc. ECCV*, 2020.

[13] Wang Bing, Lu Chen, and Bo Yang. Dm-nerf: 3d scene geometry decomposition and manipulation from 2d images. *arXiv preprint arXiv:2208.07227*, 2022.

[14] Mark Boss, Raphael Braun, Varun Jampani, Jonathan T Barron, Ce Liu, and Hendrik Lensch. Nerd: Neural reflectance decomposition from image collections. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12684–12694, 2021.

[15] Aljaž Božič, Pablo Palafox, Justus Thies, Angela Dai, and Matthias Nießner. Transformerfusion: Monocular rgb scene reconstruction using transformers. *Proc. Neural Information Processing Systems (NeurIPS)*, 2021.

[16] Alexander M Bronstein, Michael M Bronstein, Leonidas J Guibas, and Maks Ovsjanikov. Shape google: Geometric words and expressions for invariant shape retrieval. *ACM Transactions on Graphics (TOG)*, 30(1):1–20, 2011.

[17] Fabio M Carlucci, Antonio D'Innocente, Silvia Bucci, Barbara Caputo, and Tatiana Tommasi. Domain generalization by solving jigsaw puzzles. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2229–2238, 2019.

[18] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021.

[19] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. *3DV*, 2017.

[20] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong

Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015.

[21] David Charatan, Sizhe Li, Andrea Tagliasacchi, and Vincent Sitzmann. pixelsplat: 3d gaussian splats from image pairs for scalable generalizable 3d reconstruction, 2024.

[22] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields, 2022.

[23] 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. *arXiv preprint arXiv:2103.15595*, 2021.

[24] Ding-Yun Chen, Xiao-Pei Tian, Yu-Te Shen, and Ming Ouhyoung. On visual similarity based 3d model retrieval. *Computer Graphics Forum*, 22(3):223–232, 2003.

[25] Eric Ming Chen, Sidhanth Holalkere, Ruyu Yan, Kai Zhang, and Abe Davis. Ray conditioning: Trading photo-consistency for photo-realism in multi-view image generation, 2023.

[26] Guangyan Chen, Meiling Wang, Yi Yang, Kai Yu, Li Yuan, and Yufeng Yue. Pointgpt: Auto-regressively generative pre-training from point clouds. *arXiv preprint arXiv:2305.11487*, 2023.

[27] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017.

[28] Rui Chen, Songfang Han, Jing Xu, and Hao Su. Point-based multi-view stereo network. In *Proc. ICCV*, 2019.

[29] Yuedong Chen, Haofei Xu, Chuanxia Zheng, Bohan Zhuang, Marc Pollefeys, Andreas Geiger, Tat-Jen Cham, and Jianfei Cai. Mvsplat: Efficient 3d gaussian splatting from sparse multi-view images, 2024.

[30] Zhaoxi Chen, Guangcong Wang, and Ziwei Liu. Text2light: Zero-shot text-driven hdr panorama generation. *ACM TOG*, 41(6):1–16, 2022.

[31] Bowen Cheng, Maxwell D Collins, Yukun Zhu, Ting Liu, Thomas S Huang, Hartwig Adam, and Liang-Chieh Chen. Panoptic-deeplab: A simple, strong, and fast baseline for bottom-up panoptic segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12475–12485, 2020.

[32] Shuo Cheng, Zexiang Xu, Shilin Zhu, Zhuwen Li, Li Erran Li, Ravi Ramamoorthi, and Hao Su. Deep stereo using adaptive thin volume representation with uncertainty awareness. In *Proceedings of the CVPR*, pages 2524–2534, 2020.

[33] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[34] 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.

[35] Özgün Çiçek, Ahmed Abdulkadir, Soeren S Lienkamp, Thomas Brox, and Olaf Ronneberger. 3d u-net: learning dense volumetric segmentation from sparse annotation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2016: 19th International Conference, Athens, Greece, October 17-21, 2016, Proceedings, Part II 19*, pages 424–432. Springer, 2016.

[36] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017.

[37] Angela Dai and Matthias Nießner. 3dmv: Joint 3d-multi-view prediction for 3d semantic scene segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[38] Mohammad Reza Karimi Dastjerdi, Yannick Hold-Geoffroy, Jonathan Eisenmann, Siavash Khodadadeh, and Jean-François Lalonde. Guided co-modulated gan for 360° field of view extrapolation. In *3DV*, pages 475–485. IEEE, 2022.

[39] Xin Deng, WenYu Zhang, Qing Ding, and XinMing Zhang. Pointvector: A vector representation in point cloud analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9455–9465, 2023.

[40] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description, 2018.

[41] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2021.

[42] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *CVPR*, pages 12873–12883, 2021.

[43] Chuan Fang, Xiaotao Hu, Kunming Luo, and Ping Tan. Ctrl-room: Controllable text-to-3d room meshes generation with layout constraints. *arXiv preprint arXiv:2310.03602*, 2023.

[44] Mengyang Feng, Jinlin Liu, Miaomiao Cui, and Xuansong Xie. Diffusion360: Seamless 360 degree panoramic image generation based on diffusion models. *arXiv preprint arXiv:2311.13141*, 2023.

[45] Fridovich-Keil and Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *CVPR*, 2022.

[46] Kyle Genova, Forrester Cole, Avneesh Sud, Aaron Sarna, and Thomas Funkhouser. Local deep implicit functions for 3d shape. In *CVPR*, pages 4857–4866, 2020.

[47] Kyle Genova, Forrester Cole, Daniel Vlasic, Aaron Sarna, William T Freeman, and Thomas Funkhouser. Learning shape templates with structured implicit functions. In *ICCV*, pages 7154–7164, 2019.

[48] Kyle Genova, Xiaoqi Yin, Abhijit Kundu, Caroline Pantofaru, Forrester Cole, Avneesh Sud, Brian Brewington, Brian Shucker, and Thomas Funkhouser. Learning 3d semantic segmentation with only 2d image supervision. In *2021 International Conference on 3D Vision (3DV)*, pages 361–372. IEEE, 2021.

[49] Ben Graham. Sparse 3d convolutional neural networks. *arXiv preprint arXiv:1505.02890*, 2015.

[50] Xiaodong Gu, Zhiwen Fan, Siyu Zhu, Zuozhuo Dai, Feitong Tan, and Ping Tan. Cascade cost volume for high-resolution multi-view stereo and stereo matching. In *Proceedings of the CVPR*, pages 2495–2504, 2020.

[51] Abdullah Hamdi, Silvio Giancola, and Bernard Ghanem. Mvtn: Multi-view transformation network for 3d shape recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1–11, 2021.

[52] Rana Hanocka, Amir Hertz, Noa Fish, Raja Giryes, Shachar Fleishman, and Daniel Cohen-Or. Meshcnn: a network with an edge. *ACM Transactions on Graphics (TOG)*, 38(4):1–12, 2019.

[53] Takayuki Hara, Yusuke Mukuta, and Tatsuya Harada. Spherical image generation from a single image by considering scene symmetry. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 1513–1521, 2021.

[54] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022.

[55] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[56] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

[57] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[58] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[59] Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. Deep blending for free-viewpoint image-based rendering. *ACM Transactions on Graphics*, 37(6):1–15, 2018.

[60] Peter Hedman, Pratul P Srinivasan, Ben Mildenhall, Jonathan T Barron, and Paul Debevec. Baking neural radiance fields for real-time view synthesis. *arXiv preprint arXiv:2103.14645*, 2021.

[61] Alexander Hermans, Georgios Floros, and Bastian Leibe. Dense 3d semantic mapping of indoor scenes from rgb-d images. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2631–2638. IEEE, 2014.

[62] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *NeurIPS*, volume 33, pages 6840–6851, 2020.

[63] Jonathan Ho and Tim Salimans. Classifier-Free Diffusion Guidance. *arXiv:2207.12598*, 2022.

[64] Lukas Höllein, Ang Cao, Andrew Owens, Justin Johnson, and Matthias Nießner. Text2room: Extracting textured 3d meshes from 2d text-to-image models. In *ICCV*, pages 7909–7920, October 2023.

[65] 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. *International Conference on Learning Representations*, 2024.

[66] Ching-Yu Hsu, Cheng Sun, and Hwann-Tzong Chen. Moving in a 360 world: Synthesizing panoramic parallaxes from a single panorama. *CoRR*, abs/2106.10859, 2021.

[67] 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.

[68] Jingwei Huang, Haotian Zhang, Li Yi, Thomas Funkhouser, Matthias Nießner, and Leonidas J Guibas. Texturenet: Consistent local parametrizations for learning from high-resolution signals on meshes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4440–4449, 2019.

[69] Po-Han Huang, Kevin Matzen, Johannes Kopf, Narendra Ahuja, and Jia-Bin Huang. Deepmvs: Learning multi-view stereopsis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2821–2830, 2018.

[70] Yukun Huang, Jianan Wang, Yukai Shi, Boshi Tang, Xianbiao Qi, and Lei Zhang. Dreamtime: An improved optimization strategy for diffusion-guided 3d generation, 2024.

[71] 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.

[72] Rasmus Jensen, Anders Dahl, George Vogiatzis, Engil Tola, and Henrik Aanæs. Large scale multi-view stereopsis evaluation. In *2014 CVPR*, pages 406–413. IEEE, 2014.

[73] Rasmus Jensen, Anders Dahl, George Vogiatzis, Engin Tola, and Henrik Aanæs. Large scale multi-view stereopsis evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 406–413, 2014.

[74] Mengqi Ji, Juergen Gall, Haitian Zheng, Yebin Liu, and Lu Fang. SurfaceNet: An end-to-end 3D neural network for multiview stereopsis. In *Proc. ICCV*, 2017.

[75] Abhishek Kar, Christian Häne, and Jitendra Malik. Learning a multi-view stereo machine. *Advances in neural information processing systems*, 30, 2017.

[76] Abhishek Kar, Christian Häne, and Jitendra Malik. Learning a multi-view stereo machine. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 364–375, 2017.

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

[78] 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.

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

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

[81] Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *Advances in neural information processing systems*, 34:21696–21707, 2021.

[82] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollár. Panoptic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9404–9413, 2019.

[83] Sosuke Kobayashi, Eiichi Matsumoto, and Vincent Sitzmann. Decomposing nerf for editing via feature field distillation. *NIPS 2022*, 2022.

[84] Georgios Kopanas, Julien Philip, Thomas Leimkühler, and George Drettakis. Point-based neural rendering with per-view optimization. In *Computer Graphics Forum*, volume 40, pages 29–43. Wiley Online Library, 2021.

[85] Abhijit Kundu, Kyle Genova, Xiaoqi Yin, Alireza Fathi, Caroline Pantofaru, Leonidas J Guibas, Andrea Tagliasacchi, Frank Dellaert, and Thomas Funkhouser. Panoptic neural fields: A semantic object-aware neural scene representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12871–12881, 2022.

[86] Abhijit Kundu, Yin Li, Frank Dellaert, Fuxin Li, and James M Rehg. Joint semantic segmentation and 3d reconstruction from monocular video. In *European Conference on Computer Vision*, pages 703–718. Springer, 2014.

[87] Abhijit Kundu, Xiaoqi Yin, Alireza Fathi, David Ross, Brian Brewington, Thomas Funkhouser, and Caroline Pantofaru. Virtual multi-view fusion for 3d semantic segmentation. In *European Conference on Computer Vision*, pages 518–535. Springer, 2020.

[88] Black Forest Labs. flux. https://github.com/black-forest-labs/flux, 2024.

[89] Christoph Lassner and Michael Zollhofer. Pulsar: Efficient sphere-based neural rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1440–1449, 2021.

[90] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. *Advances in neural information processing systems*, 31, 2018.

[91] Yangyan Li, Hao Su, Charles Ruizhongtai Qi, Noa Fish, Daniel Cohen-Or, and Leonidas J Guibas. Joint embeddings of shapes and images via cnn image purification. *ACM transactions on graphics (TOG)*, 34(6):1–12, 2015.

[92] Yuyan Li, Yuliang Guo, Zhixin Yan, Xinyu Huang, Ye Duan, and Liu Ren. Omnifusion: 360 monocular depth estimation via geometry-aware fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2801–2810, 2022.

[93] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6498–6508, 2021.

[94] Yiyi Liao, Jun Xie, and Andreas Geiger. KITTI-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d. *arXiv preprint arXiv:2109.13410*, 2021.

[95] Haojia Lin, Xiawu Zheng, Lijiang Li, Fei Chao, Shanshan Wang, Yan Wang, Yonghong Tian, and Rongrong Ji. Meta architecure for point cloud analysis. *arXiv:2211.14462*, 2022.

[96] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[97] Jingguo Liu, Yijun Xu, Shigang Li, and Jianfeng Li. Estimating depth of monocular panoramic image with teacher-student model fusing equirectangular and spherical representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1262–1271, 2024.

[98] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *arXiv preprint arXiv:2007.11571*, 2020.

[99] 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.

[100] 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 *Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2023.

[101] 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, 2023.

[102] 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.

[103] 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. *International Conference on Computer Vision*, 2023.

[104] 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.

[105] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021.

[106] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *arXiv preprint arXiv:1906.07751*, 2019.

[107] Stephen Lombardi, Tomas Simon, Gabriel Schwartz, Michael Zollhoefer, Yaser Sheikh, and Jason Saragih. Mixture of volumetric primitives for efficient neural rendering. *ACM Transactions on Graphics (TOG)*, 40(4):1–13, 2021.

[108] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[109] Zhuqiang Lu, Kun Hu, Chaoyue Wang, Lei Bai, and Zhiyong Wang. Autoregressive omni-aware outpainting for open-vocabulary 360-degree image generation. *arXiv preprint arXiv:2309.03467*, 2023.

[110] Keyang Luo, Tao Guan, Lili Ju, Haipeng Huang, and Yawei Luo. P-mvsnet: Learning patch-wise matching confidence aggregation for multi-view stereo. In *Proceedings of the ICCV*, pages 10452–10461, 2019.

[111] Lingni Ma, Jörg Stückler, Christian Kerl, and Daniel Cremers. Multi-view deep learning for consistent semantic mapping with rgb-d cameras. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 598–605. IEEE, 2017.

[112] 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.

[113] Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7210–7219, 2021.

[114] John McCormac, Ankur Handa, Andrew Davison, and Stefan Leutenegger. Semanticfusion: Dense 3d semantic mapping with convolutional neural networks. In *2017 IEEE International Conference on Robotics and automation (ICRA)*, pages 4628–4635. IEEE, 2017.

[115] 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 *European conference on computer vision*, pages 405–421. Springer, 2020.

[116] 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. *Communications of the ACM*, 65(1):99–106, 2021.

[117] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, July 2022.

[118] Gaku Narita, Takashi Seno, Tomoya Ishikawa, and Yohsuke Kaji. Panopticfusion: Online volumetric semantic mapping at the level of stuff and things. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4205–4212. IEEE, 2019.

[119] Alexey Nekrasov, Jonas Schult, Or Litany, Bastian Leibe, and Francis Engelmann. Mix3d: Out-of-context data augmentation for 3d scenes. In *2021 International Conference on 3D Vision (3DV)*, pages 116–125. IEEE, 2021.

[120] Richard A Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *2011 10th IEEE international symposium on mixed and augmented reality*, pages 127–136. IEEE, 2011.

[121] Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Marc Stamminger. Real-time 3d reconstruction at scale using voxel hashing. *ACM Transactions on Graphics (ToG)*, 32(6):1–11, 2013.

[122] Changgyoon Oh, Wonjune Cho, Yujeong Chae, Daehee Park, Lin Wang, and Kuk-Jin Yoon. Bips: Bi-modal indoor panorama synthesis via residual depth-aided adversarial learning. In *ECCV*, pages 352–371. Springer, 2022.

[123] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

[124] Robert Osada, Thomas Funkhouser, Bernard Chazelle, and David Dobkin. Matching 3d models with shape distributions. In *Proceedings International Conference on Shape Modeling and Applications*, pages 154–166. IEEE, 2001.

[125] Julian Ost, Fahim Mannan, Nils Thuerey, Julian Knodt, and Felix Heide. Neural scene graphs for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2856–2865, 2021.

[126] 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.

[127] Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5865–5874, 2021.

[128] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M Seitz. Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields. *arXiv preprint arXiv:2106.13228*, 2021.

[129] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2536–2544, 2016.

[130] Songyou Peng, Kyle Genova, Chiyu Jiang, Andrea Tagliasacchi, Marc Pollefeys, Thomas Funkhouser, et al. Openscene: 3d scene understanding with open vocabularies. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 815–824, 2023.

[131] Quang-Hieu Pham, Thanh Nguyen, Binh-Son Hua, Gemma Roig, and Sai-Kit Yeung. Jsis3d: joint semantic-instance segmentation of 3d point clouds with multi-task pointwise networks and multi-value conditional random fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8827–8836, 2019.

[132] Matia Pizzoli, Christian Forster, and Davide Scaramuzza. Remode: Probabilistic, monocular dense reconstruction in real time. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2609–2616. IEEE, 2014.

[133] Julius Plucker. Xvii. on a new geometry of space. *Philosophical Transactions of the Royal Society of London*, pages 725–791, 1865.

[134] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. Dreamfusion: Text-to-3D using 2D diffusion. *arXiv*, 2022.

[135] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proc. CVPR*, 2017.

[136] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in neural information processing systems*, pages 5099–5108, 2017.

[137] 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.

[138] Guocheng Qian, Yuchen Li, Houwen Peng, Jinjie Mai, Hasan Hammoud, Mohamed Elhoseiny, and Bernard Ghanem. Pointnext: Revisiting pointnet++ with improved training and scaling strategies. *Advances in Neural Information Processing Systems*, 35:23192–23204, 2022.

[139] 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.

[140] 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.

[141] Haoxi Ran, Jun Liu, and Chengjie Wang. Surface representation for point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18942–18952, 2022.

[142] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14335–14345, 2021.

[143] Google Research. Google scanned objects.

[144] Tal Ridnik, Emanuel Ben-Baruch, Asaf Noy, and Lihi Zelnik-Manor. Imagenet-21k pretraining for the masses. *arXiv preprint arXiv:2104.10972*, 2021.

[145] Gernot Riegler and Vladlen Koltun. Free view synthesis. In *ECCV*, 2020.

[146] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. Octnet: Learning deep 3d representations at high resolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3577–3586, 2017.

[147] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[148] Darius Rückert, Linus Franke, and Marc Stamminger. Adop: Approximate differentiable one-pixel point rendering. *arXiv preprint arXiv:2110.06635*, 2021.

[149] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[150] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise View Selection for Unstructured Multi-View Stereo. In *European Conference on Computer Vision (ECCV)*, 2016.

[151] Jonas Schult, Francis Engelmann, Alexander Hermans, Or Litany, Siyu Tang, and Bastian Leibe. Mask3d for 3d semantic instance segmentation. *arXiv preprint arXiv:2210.03105*, 2022.

[152] Steven M Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *2006 IEEE computer society conference on CVPR*, volume 1, pages 519–528. IEEE, 2006.

[153] 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.

[154] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. *arXiv preprint arXiv:1912.13192*, 2019.

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

[156] Karen Simonyan. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[157] Gowri Somanath and Daniel Kurz. Hdr environment map estimation for real-time augmented reality. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11298–11306, 2021.

[158] Liangchen Song, Liangliang Cao, Hongyu Xu, Kai Kang, Feng Tang, Junsong Yuan, and Zhao Yang. Roomdreamer: Text-driven 3d indoor scene synthesis with coherent geometry and texture. In Abdulmotaleb El-Saddik, Tao Mei, Rita Cucchiara, Marco Bertini, Diana Patricia Tobon Vallejo, Pradeep K. Atrey, and M. Shamim Hossain, editors, *ACM MM*, pages 6898–6906. ACM, 2023.

[159] Shuran Song, Fisher Yu, Andy Zeng, Angel X Chang, Manolis Savva, and Thomas Funkhouser. Semantic scene completion from a single depth image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1746–1754, 2017.

[160] Pratul P. Srinivasan, Boyang Deng, Xiuming Zhang, Matthew Tancik, Ben Mildenhall, and Jonathan T. Barron. Nerv: Neural reflectance and visibility fields for relighting and view synthesis. In *CVPR*, 2021.

[161] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J. Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, Anton Clarkson, Mingfei Yan, Brian Budge, Yajie Yan, Xiaqing Pan, June Yon, Yuyang Zou, Kimberly Leon, Nigel Carter, Jesus Briales, Tyler Gillingham, Elias Mueggler, Luis Pesqueira, Manolis Savva, Dhruv Batra, Hauke M. Strasdat, Renzo De Nardi, Michael Goesele, Steven Lovegrove, and Richard Newcombe. The Replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019.

[162] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *CVPR*, 2022.

[163] Jiaming Sun, Yiming Xie, Linghao Chen, Xiaowei Zhou, and Hujun Bao. NeuralRecon: Real-time coherent 3D reconstruction from monocular video. *CVPR*, 2021.

[164] Stanislaw Szymanowicz, Eldar Insafutdinov, Chuanxia Zheng, Dylan Campbell, João F. Henriques, Christian Rupprecht, and Andrea Vedaldi. Flash3d: Feed-forward generalisable 3d scene reconstruction from a single image, 2024.

[165] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le. Mnasnet: Platform-aware neural architecture search for mobile. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2820–2828, 2019.

[166] 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 *European Conference on Computer Vision (ECCV)*, 2020.

[167] Shitao Tang, Fuyang Zhang, Jiacheng Chen, Peng Wang, and Yasutaka Furukawa. Mvdiffusion: Enabling holistic multi-view image generation with correspondence-aware diffusion. In *NeurIPS*, 2023.

[168] 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 International Conference on Computer Vision*, pages 6411–6420, 2019.

[169] Haithem Turki, Deva Ramanan, and Mahadev Satyanarayanan. Mega-nerf: Scalable construction of large-scale nerfs for virtual fly-throughs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12922–12931, 2022.

[170] Mikaela Angelina Uy, Quang-Hieu Pham, Binh-Son Hua, Duc Thanh Nguyen, and Sai-Kit Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In *International Conference on Computer Vision (ICCV)*, 2019.

[171] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[172] Vibhav Vineet, Ondrej Miksik, Morten Lidegaard, Matthias Nießner, Stuart Golodetz, Victor A Prisacariu, Olaf Kähler, David W Murray, Shahram Izadi, Patrick Pérez, et al. Incremental dense semantic stereo fusion for large-scale semantic scene reconstruction. In *2015 IEEE international conference on robotics and automation (ICRA)*, pages 75–82. IEEE, 2015.

[173] Suhani Vora, Noha Radwan, Klaus Greff, Henning Meyer, Kyle Genova, Mehdi SM Sajjadi, Etienne Pot, Andrea Tagliasacchi, and Daniel Duckworth. NeSF: Neural semantic fields for generalizable semantic segmentation of 3d scenes. *arXiv preprint arXiv:2111.13260*, 2021.

[174] Suhani Vora, Noha Radwan, Klaus Greff, Henning Meyer, Kyle Genova, Mehdi SM Sajjadi, Etienne Pot, Andrea Tagliasacchi, and Daniel Duckworth. Nesf: Neural semantic fields for generalizable semantic segmentation of 3d scenes. *arXiv preprint arXiv:2111.13260*, 2021.

[175] 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.

[176] Guangcong Wang, Yinuo Yang, Chen Change Loy, and Ziwei Liu. Stylelight: Hdr panorama generation for lighting estimation and editing. In *ECCV*, pages 477–492. Springer, 2022.

[177] Hai Wang, Xiaoyu Xiang, Yuchen Fan, and Jing-Hao Xue. Customizing 360-degree panoramas through text-to-image diffusion models. In *WACV*, 2024.

[178] Haiyang Wang, Shaocong Dong, Shaoshuai Shi, Aoxue Li, Jianan Li, Zhenguo Li, Liwei Wang, et al. Cagroup3d: Class-aware grouping for 3d object detection on point clouds. *Advances in Neural Information Processing Systems*, 35:29975–29988, 2022.

[179] Jionghao Wang, Ziyu Chen, Jun Ling, Rong Xie, and Li Song. 360-degree panorama generation from few unregistered nfov images. In Abdulmotaleb El-Saddik, Tao Mei, Rita Cucchiara, Marco Bertini, Diana Patricia Tobon Vallejo, Pradeep K. Atrey, and M. Shamim Hossain, editors, *ACM MM*, pages 6811–6821. ACM, 2023.

[180] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul Srinivasan, Howard Zhou, Jonathan T. Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. Ibrnet: Learning multi-view image-based rendering. In *CVPR*, 2021.

[181] 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. In *NeurIPS*, 2023.

[182] 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, 2024.

[183] Silvan Weder, Johannes Schonberger, Marc Pollefeys, and Martin R Oswald. Routedfusion: Learning real-time depth map fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4887–4897, 2020.

[184] 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, 2024.

[185] Yi Wei, Shaohui Liu, Yongming Rao, Wang Zhao, Jiwen Lu, and Jie Zhou. Nerfingmvs: Guided optimization of neural radiance fields for indoor multi-view stereo. In *ICCV*, 2021.

[186] Bo Wu, Yang Liu, Bo Lang, and Lei Huang. Dgcnn: Disordered graph convolutional neural network based on the gaussian mixture model. *Neurocomputing*, 321:346–356, 2018.

[187] Liwen Wu, Jae Yong Lee, Anand Bhattad, Yuxiong Wang, and David Forsyth. Diver: Real-time and accurate neural radiance fields with deterministic integration for volume rendering, 2021.

[188] Tianhao Wu, Chuanxia Zheng, and Tat-Jen Cham. Ipo-ldm: Depth-aided 360-degree indoor rgb panorama outpainting via latent diffusion model. *arXiv preprint arXiv:2307.03177*, 2023.

[189] Tianhao Wu, Chuanxia Zheng, and Tat-Jen Cham. Panodiffusion: 360-degree panorama outpainting via diffusion, 2024.

[190] Xiaoyang Wu, Zhuotao Tian, Xin Wen, Bohao Peng, Xihui Liu, Kaicheng Yu, and Hengshuang Zhao. Towards large-scale 3d representation learning with multi-dataset point prompt training. *arXiv preprint arXiv:2308.09718*, 2023.

[191] Xiaoyang Wu, Xin Wen, Xihui Liu, and Hengshuang Zhao. Masked scene contrast: A scalable framework for unsupervised 3d representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9415–9424, 2023.

[192] 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.

[193] Fanbo Xiang, Zexiang Xu, Milos Hasan, Yannick Hold-Geoffroy, Kalyan Sunkavalli, and Hao Su. Neutex: Neural texture mapping for volumetric neural rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7119–7128, 2021.

[194] 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.

[195] Chenfeng Xu, Bichen Wu, Ji Hou, Sam Tsai, Ruilong Li, Jialiang Wang, Wei Zhan, Zijian He, Peter Vajda, Kurt Keutzer, et al. Nerf-det: Learning geometry-aware volumetric representation for multi-view 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 23320–23330, 2023.

[196] Hu Xu, Saining Xie, Xiaoqing Ellen Tan, Po-Yao Huang, Russell Howes, Vasu Sharma, Shang-Wen Li, Gargi Ghosh, Luke Zettlemoyer, and Christoph Feichtenhofer. Demystifying clip data, 2023.

[197] Yinghao Xu, Zifan Shi, Wang Yifan, Hansheng Chen, Ceyuan Yang, Sida Peng, Yujun Shen, and Gordon Wetzstein. Grm: Large gaussian reconstruction model for efficient 3d reconstruction and generation, 2024.

[198] Yinghao Xu, Hao Tan, Fujun Luan, Sai Bi, Peng Wang, Jiahao Li, Zifan Shi, Kalyan Sunkavalli, Gordon Wetzstein, Zexiang Xu, and Kai Zhang. Dmv3d: Denoising multi-view diffusion using 3d large reconstruction model, 2023.

[199] Le Xue, Mingfei Gao, Chen Xing, Roberto Martín-Martín, Jiajun Wu, Caiming Xiong, Ran Xu, Juan Carlos Niebles, and Silvio Savarese. Ulip: Learning a unified representation of language, images, and point clouds for 3d understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1179–1189, 2023.

[200] Le Xue, Ning Yu, Shu Zhang, Junnan Li, Roberto Martín-Martín, Jiajun Wu, Caiming Xiong, Ran Xu, Juan Carlos Niebles, and Silvio Savarese. Ulip-2: Towards scalable multimodal pre-training for 3d understanding. *arXiv preprint arXiv:2305.08275*, 2023.

[201] Bangbang Yang, Wenqi Dong, Lin Ma, Wenbo Hu, Xiao Liu, Zhaopeng Cui, and Yuewen Ma. Dreamspace: Dreaming your room space with text-driven panoramic texture propagation. *arXiv preprint arXiv:2310.13119*, 2023.

[202] Bangbang Yang, Yinda Zhang, Yinghao Xu, Yijin Li, Han Zhou, Hujun Bao, Guofeng Zhang, and Zhaopeng Cui. Learning object-compositional neural radiance field for editable scene rendering. In *International Conference on Computer Vision (ICCV)*, October 2021.

[203] 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. In *Advances in Neural Information Processing Systems*, pages 6737–6746, 2019.

[204] Yu-Qi Yang, Yu-Xiao Guo, Jian-Yu Xiong, Yang Liu, Hao Pan, Peng-Shuai Wang, Xin Tong, and Baining Guo. Swin3d: A pretrained transformer backbone for 3d indoor scene understanding. *arXiv preprint arXiv:2304.06906*, 2023.

[205] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. MVSnet: Depth inference for unstructured multi-view stereo. In *Proc. ECCV*, pages 767–783, 2018.

[206] Yao Yao, Zixin Luo, Shiwei Li, Tianwei Shen, Tian Fang, and Long Quan. Recurrent mvsnet for high-resolution multi-view stereo depth inference. In *Proceedings of the CVPR*, pages 5525–5534, 2019.

[207] Jianglong Ye, Naiyan Wang, and Xiaolong Wang. Featurenerf: Learning generalizable nerfs by distilling foundation models. *arXiv preprint arXiv:2303.12786*, 2023.

[208] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *CVPR*, 2021.

[209] 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.

[210] Jason J. Yu, Fereshteh Forghani, Konstantinos G. Derpanis, and Marcus A. Brubaker. Long-term photometric consistent novel view synthesis with diffusion models. In *ICCV*, 2023.

[211] Xianggang Yu, Mutian Xu, Yidan Zhang, Haolin Liu, Chongjie Ye, Yushuang Wu, Zizheng Yan, Tianyou Liang, Guanying Chen, Shuguang Cui, and Xiaoguang Han. Mvimgnet: A large-scale dataset of multi-view images. In *CVPR*, 2023.

[212] 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.

[213] Karim Abou Zeid, Jonas Schult, Alexander Hermans, and Bastian Leibe. Point2vec for self-supervised representation learning on point clouds. *arXiv preprint arXiv:2303.16570*, 2023.

[214] Cheng Zhang, Zhi Liu, Guangwen Liu, and Dandan Huang. Large-scale 3d semantic mapping using monocular vision. In *2019 IEEE 4th International Conference on Image, Vision and Computing (ICIVC)*, pages 71–76. IEEE, 2019.

[215] Cheng Zhang, Qianyi Wu, Camilo Cruz Gambardella, Xiaoshui Huang, Dinh Phung, Wanli Ouyang, and Jianfei Cai. Taming stable diffusion for text to 360-degree panorama image generation, 2024.

[216] Chubin Zhang, Hongliang Song, Yi Wei, Yu Chen, Jiwen Lu, and Yansong Tang. Geolrm: Geometry-aware large reconstruction model for high-quality 3d gaussian generation, 2024.

[217] Hang Zhang, Chongruo Wu, Zhongyue Zhang, Yi Zhu, Haibin Lin, Zhi Zhang, Yue Sun, Tong He, Jonas Mueller, R Manmatha, et al. Resnest: Split-attention networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2736–2746, 2022.

[218] Kai Zhang, Sai Bi, Hao Tan, Yuanbo Xiangli, Nanxuan Zhao, Kalyan Sunkavalli, and Zexiang Xu. Gs-lrm: Large reconstruction model for 3d gaussian splatting, 2024.

[219] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. Nerf++: Analyzing and improving neural radiance fields. *arXiv preprint arXiv:2010.07492*, 2020.

[220] 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.

[221] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III 14*, pages 649–666. Springer, 2016.

[222] 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.

[223] Xiaoshuai Zhang, Abhijit Kundu, Thomas Funkhouser, Leonidas Guibas, Hao Su, and Kyle Genova. Nerflets: Local radiance fields for efficient structure-aware 3d scene representation from 2d supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8274–8284, 2023.

[224] 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.

[225] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *CVPR*, 2017.

[226] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip HS Torr, et al. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6881–6890, 2021.

[227] Shuaifeng Zhi, Tristan Laidlow, Stefan Leutenegger, and Andrew Davison. In-Place Scene Labelling and Understanding with Implicit Scene Representation. In *ICCV*, 2021.

[228] Shuaifeng Zhi, Tristan Laidlow, Stefan Leutenegger, and Andrew Davison. In-place scene labelling and understanding with implicit scene representation. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021.

[229] Shuaifeng Zhi, Tristan Laidlow, Stefan Leutenegger, and Andrew J. Davison. In-place scene labelling and understanding with implicit scene representation. In *ICCV*, 2021.

[230] Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning deep features for scene recognition using places database. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

[231] Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. ibot: Image bert pre-training with online tokenizer. *arXiv preprint arXiv:2111.07832*, 2021.

[232] Junsheng Zhou, Jinsheng Wang, Baorui Ma, Yu-Shen Liu, Tiejun Huang, and Xinlong Wang. Uni3d: Exploring unified 3d representation at scale. *arXiv preprint arXiv:2310.06773*, 2023.

[233] Qian-Yi Zhou and Vladlen Koltun. Color map optimization for 3D reconstruction with consumer depth cameras. *ACM Transactions on Graphics*, 33(4):155, 2014.

[234] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 37, 2018.

[235] Haoyi Zhu, Honghui Yang, Xiaoyang Wu, Di Huang, Sha Zhang, Xianglong He, Tong He, Hengshuang Zhao, Chunhua Shen, Yu Qiao, et al. Ponderv2: Pave the way for 3d foundataion model with a universal pre-training paradigm. *arXiv preprint arXiv:2310.08586*, 2023.

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