Thèse n° 10 641

Advancing Self-Supervised Deep Learning for 3D Scene Understanding

Présentée le 30 mai 2024

Faculté des sciences et techniques de l'ingénieur
Laboratoire de l'IDIAP
Programme doctoral en génie électrique

pour l'obtention du grade de Docteur ès Sciences

par

Seyed Mohammad Mahdi JOHARI

École
polytechnique
fédérale
de Lausanne

2024

Acknowledgements

The completion of this thesis would not have been possible without the unwavering support and guidance of numerous individuals who have played a significant role in my academic journey.

First and foremost, I extend my deepest gratitude to my supervisor, Prof. François Fleuret, for their unwavering mentorship and belief in my abilities. Their insightful guidance, constructive feedback, and relentless encouragement have been instrumental in shaping my research and ensuring its clarity and rigor. I am deeply indebted to their expertise and dedication.

Secondly, I am deeply appreciative of the insightful comments, intellectually stimulating discussions, and generous sharing of knowledge and expertise provided by the researchers and coauthors at ams OSRAM, Hugues Salamin, Camilla Carta, and Yann Lepoittevin. Their insights and collaborative spirit were invaluable in enriching my research experience and expanding my understanding of the subject matter.

In addition, I would like to express my sincere gratitude to the entire Idiap team, in particular, Prof. Daniel Gatica-Perez, for creating a supportive and enriching environment for my PhD studies.

I am also thankful to Jan Rüegg and Sebastian Martin from Apple Zurich, whose guidance and collaboration during my internship profoundly enriched my proficiency in 3D rendering and familiarized me with the intricacies of the industry environment.

Last but not least, I extend my heartfelt thanks to my family and friends for their unwavering love, support, and encouragement throughout this challenging yet rewarding journey. Their belief in me served as a constant source of motivation, and their unwavering support helped me persevere through moments of doubt and self-doubt.

Lausanne, April 24, 2024 S. Johari

Abstract

Recent advancements in deep learning have revolutionized 3D computer vision, enabling the extraction of intricate 3D information from 2D images and video sequences. This thesis explores the application of deep learning in three crucial challenges of 3D computer vision: Depth Estimation, Novel View Synthesis, and Simultaneous Localization and Mapping (SLAM).

In the first part of the study, a self-supervised deep-learning method for depth estimation using a structured-light camera is proposed. Our method utilizes optical flow for improved edge preservation and reduced over-smoothing. In addition, we propose fusing depth maps from multiple video frames to enhance overall accuracy, particularly in occluded areas. Further, we demonstrate that these fused depth maps can be used for self-supervision to further improve the performance of a single-frame depth estimation network. Our models outperform state-of-the-art methods on both synthetic and real datasets.

In the second part of the study, a generalizable photorealistic novel view synthesis method based on neural radiance fields (NeRF) is introduced. Our approach employs a geometry reasoner and a renderer to generate high-quality images from novel viewpoints. The geometry reasoner constructs cascaded cost volumes for each nearby source view, while the renderer utilizes a Transformer-based attention mechanism to integrate information from these cost volumes and render detailed images using volume rendering techniques. This architecture enables sophisticated occlusion reasoning and allows our method to render competitive results with per-scene optimized neural rendering methods while significantly reducing computational costs. Our experiments demonstrate superiority over state-of-the-art generalizable neural rendering models on various synthetic and real datasets.

In the last part of the study, an efficient implicit neural representation method for dense visual SLAM is presented. The method reconstructs the scene representation while simultaneously estimating the camera position in a sequential manner from RGB-D frames with unknown poses. We incorporate recent advances in NeRF into the SLAM system, achieving both high accuracy and efficiency. The scene representation consists of multi-scale axis-aligned perpendicular feature planes and shallow decoders that decode the interpolated features into Truncated Signed Distance Field (TSDF) and RGB values. Extensive experiments on standard datasets demonstrate that our method outperforms state-of-the-art dense visual SLAM methods by more than 50% in 3D reconstruction and camera localization while running up to 10 times faster and eliminating the need for pre-training.

Keywords: deep learning, 3D computer vision, depth estimation, novel view synthesis, neural radiance fields (NeRF), scene reconstruction, simultaneous localization and mapping (SLAM)

Résumé

Les récentes avancées dans le domaine de l'apprentissage profond ont révolutionné la vision 3D par ordinateur, permettant l'extraction d'informations 3D complexes à partir d'images 2D et de séquences vidéo. Cette thèse explore l'application de l'apprentissage profond à trois défis cruciaux de la vision 3D par ordinateur : L'estimation de la profondeur (Depth Estimation), la synthèse de nouvelles vues (Novel View Synthesis) et la localisation et la cartographie simultanées (SLAM).

Dans la première partie de l'étude, une méthode d'apprentissage profond auto-supervisée pour l'estimation de la profondeur à l'aide d'une caméra à lumière structurée est proposée. Notre méthode utilise le flux optique pour une meilleure préservation des bords et un lissage excessif réduit. En outre, nous proposons de fusionner les cartes de profondeur de plusieurs images vidéo afin d'améliorer la précision globale, en particulier dans les zones occultées. En outre, nous démontrons que ces cartes de profondeur fusionnées peuvent être utilisées pour l'auto-supervision afin d'améliorer encore les performances d'un réseau d'estimation de profondeur à image unique. Nos modèles sont plus performants que les méthodes les plus récentes sur des ensembles de données synthétiques et réelles.

Dans la deuxième partie de l'étude, une méthode généralisable de synthèse photoréaliste de nouvelles vues basée sur les champs de radiance neuronaux (NeRF) est introduite. Notre approche utilise un raisonneur géométrique et un moteur de rendu pour générer des images de haute qualité à partir de nouveaux points de vue. Le raisonneur géométrique construit des volumes de coûts en cascade pour chaque vue source proche, tandis que le moteur de rendu utilise un mécanisme d'attention basé sur un transformateur pour intégrer les informations de ces volumes de coûts et rendre des images détaillées en utilisant des techniques de rendu de volume. Cette architecture permet un raisonnement sophistiqué en matière d'occlusion et permet à notre méthode d'obtenir des résultats compétitifs par rapport aux méthodes de rendu neuronal optimisées par scène, tout en réduisant considérablement les coûts de calcul. Nos expériences démontrent la supériorité de notre méthode par rapport aux modèles de rendu neuronal généralisables les plus récents sur divers ensembles de données synthétiques et réelles.

Dans la dernière partie de l'étude, une méthode efficace de représentation neuronale implicite pour le SLAM visuel dense est présentée. La méthode reconstruit la représentation de la scène tout en estimant simultanément la position de la caméra de manière séquentielle à partir d'images RVB-D dont les poses sont inconnues. Nous intégrons les récentes avancées en matière de NeRF dans le système SLAM, ce qui permet d'obtenir une précision et une efficacité élevées. La représentation de la scène se compose de plans de caractéristiques perpendiculaires alignés sur plusieurs échelles et de décodeurs peu profonds qui décodent les caractéristiques interpolées en champ de distance signé tronqué (TSDF) et en valeurs RVB. Des expériences approfondies sur des ensembles de données standard démontrent que notre méthode surpasse les méthodes SLAM visuelles denses de plus de 50 % dans la reconstruction 3D et la localisation de la caméra, tout en fonctionnant jusqu'à 10 fois plus vite et en éliminant le besoin de pré-entraînement.

Mots-clés : apprentissage profond, vision par ordinateur 3D, estimation de la profondeur, synthèse de vues nouvelles, champs de radiance neuronaux (NeRF), reconstruction de scènes, localisation et cartographie simultanées (SLAM)

Contents

List of Figures

List of Tables

Introduction Part I

1 Introduction and Thesis Overview

Disclaimer: This chapter is adapted from the following articles – with permission of all co-authors and the conference:

Johari, M. M., Carta, C., and Fleuret, F. (2021). DepthInSpace: Exploitation and Fusion of Multiple Video Frames for Structured-Light Depth Estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6039-6048.

Johari, M. M., Lepoittevin, Y., and Fleuret, F. (2022). GeoNeRF: Generalizing NeRF with Geometry Priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18365–18375.

Johari, M. M., Carta, C., and Fleuret, F. (2023). ESLAM: Efficient Dense SLAM System Based on Hybrid Representation of Signed Distance Fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17408-17419. Selected as a **Highlight Paper** (Top 2.5%).

1.1 Introduction

In recent times, the emergence of deep learning has brought about a significant transformation in the field of computer vision, marking a shift in how machines interpret visual information. The fusion of artificial intelligence and neural network structures, particularly the multi-layered configurations of deep neural networks, has fundamentally changed the landscape of visual processing. At the core of deep learning is its ability to independently learn hierarchical representations from raw data, making it a powerful tool for addressing intricate visual tasks that were previously considered insurmountable. The impact of this approach extends beyond mere image recognition, influencing various sectors such as robotics, healthcare, security, and entertainment.

In the context of deep learning for computer vision, a pivotal moment occurred in 2012 when Krizhevsky et al. (2012) introduced their work, now known as AlexNet, a groundbreaking convolutional neural network (CNN) architecture. This innovative architecture marked a new era by significantly surpassing traditional methods in the ImageNet Large Scale Visual Recognition Challenge (Russakovsky et al., 2015). This accomplishment not only showcased the effectiveness of deep learning in image classification but also ignited a substantial increase in research and development within the field. Successive architectures, including VGGNet (Simonyan and Zisserman, 2015), GoogLeNet (Szegedy et al., 2015), ResNet (He et al., 2016), and ViT (Dosovitskiy et al., 2021), played a crucial role in advancing accuracy and scalability, solidifying deep learning as the cornerstone of modern computer vision.

As we explore the vast realm of deep learning, it is crucial to acknowledge its impact on diverse applications where the ability to interpret visual information has become essential. Autonomous systems, driven by deep learning algorithms, now navigate intricate environments with unprecedented precision and adaptability (Chib and Singh, 2023). In medical imaging, deep learning aids in identifying anomalies, improving diagnostic accuracy, and potentially revolutionizing patient care (Suganyadevi et al., 2022). Biometric systems, supported by advanced neural networks, redefine security protocols (Jadhav et al., 2022), while augmented reality experiences benefit from the seamless integration of deep learning for object recognition and scene understanding (Ghasemi et al., 2022).

The interaction between deep learning and computer vision goes beyond task optimization; instead, it signifies a symbiotic relationship that continually evolves to address emerging challenges.

Deep learning has significantly impacted not only traditional computer vision tasks but has also brought about transformative changes in the field of 3D computer vision and scene understanding. The incorporation of deep learning methods into the three-dimensional domain signifies a logical progression in artificial intelligence evolution and introduces numerous possibilities to enhance machine perceptual capabilities.

Moving into the third dimension introduces a new layer of complexity, extending beyond the traditional boundaries of two-dimensional image processing. The unique nature of three-dimensional data requires advanced techniques capable of decoding spatial relationships and reconstructing scenes with a nuanced understanding of the complexities inherent in a multidimensional world. This shift necessitates models to address challenges related to depth perception, volumetric understanding, and the intricate interplay of objects within a spatial context. Deep learning, known for its capacity to autonomously learn hierarchical representations from raw data, plays a crucial role in addressing these challenges. It provides a robust framework for machines to develop a nuanced understanding of spatial structures and intricate relationships within the volumetric domain.

In the realm of 3D computer vision, challenges go beyond conventional image recognition as algorithms must now navigate and interpret the intricacies of spatial dimensions. The emphasis shifts from merely identifying present objects to determining their precise locations in three-dimensional space. This shift requires departing from traditional methodologies and encourages the development of novel approaches that leverage the power of deep neural networks to extract insights from volumetric data and multiple perspectives (Maturana and Scherer, 2015; Qi et al., 2017a; Li et al., 2018b).

In recent years, 3D computer vision has emerged as a dynamic and rapidly evolving field, revolutionizing the way machines perceive and interact with the three-dimensional world. This interdisciplinary domain intersects computer science, mathematics, and optics, aiming to replicate human-like depth perception in machines. From autonomous navigation to augmented reality applications, 3D computer vision tasks have found diverse applications across various industries. In the following, a glimpse into some examples of the prominent 3D computer vision tasks is provided.

Object Recognition and Detection. Object Recognition and Detection is a pivotal area in computer vision, which involves the identification and localization of objects within a three-dimensional space. This field has witnessed significant advancements, with methods leveraging point cloud data, depth information, and sophisticated algorithms. PointNet (Qi et al., 2017a) marked a milestone by directly processing point clouds for recognition tasks. Building on this, PointNet++ (Qi et al., 2017b) and Frustum PointNets (Qi et al., 2018) proposed incremental improvements. Recent advances in 3D object recognition and detection have been marked by the integration of deep learning and large-scale datasets. Notable contributions include Diffusion-SS3D (Ho et al., 2023), which addresses the limitation of the availability of large-scale 3D annotations by exploiting pseudo-labels, and MonoNeRD (Xu et al., 2023a), introducing a Neural Radiance Field-based object detection pipeline. These works highlight the ongoing progress in leveraging neural networks to enhance accuracy and efficiency in 3D object analysis.

Depth Estimation. Depth estimation, predicting the distance of each pixel from the camera, is crucial for scene understanding or reconstruction. Monodepth2 (Godard et al., 2019) is one of the pioneering works in the domain that proposes a robust self-supervised approach for depth estimation from a mixture of monocular and stereo images. Numerous researchers have been following this fundamental line of research in the 3D domain recently, including but not limited to the studies in Zhao et al. (2023), Shao et al. (2023), Yang et al. (2023c), Zhou and Dong (2023), and Piccinelli et al. (2023). In addition to depth sensing from RGB images, another intriguing line of research involves depth estimation from the raw data of active sensors, such as Time-of-Flight (Li et al., 2022b), LiDAR (Bartoccioni et al., 2023), and structured-light (Riegler et al., 2019).

Object or Scene Reconstruction. Creating 3D models from 2D images or RGB-D images is the aim of reconstruction approaches. Pixel2Mesh (Wang et al., 2018a) and Pix2Vox (Xie et al., 2019) introduce methods leveraging neural networks for generating 3D mesh models from a single image. More recent work like the ones by Yang et al. (2023b) and Zhang et al. (2022) attempt to improve the generalizability of the reconstruction to unseen object categories. 3D reconstruction is not limited to objects in the literature. Methods like neuralRecon (Sun et al., 2021), TransformerFusion (Bozic et al., 2021), Manhattan-SDF (Guo et al., 2022), SceneRF (Cao and de Charette, 2023), and Uni-3D (Zhang et al., 2023b) show promising result in large-scale scene reconstruction from multi-view inputs.

3D Pose Estimation. Determining the spatial configuration of objects or humans in a scene is the task of 3D pose estimation. DeepPose (Toshev and Szegedy, 2014) pioneers a deep learning approach for estimating human pose in 3D from 2D images. More recent advanced methods for pose estimation can be found in the works by Zhang et al. (2023c) and Zhou et al. (2023) which are robust to occlusions thanks to 3D understanding of the context.

Novel View Synthesis. Novel view synthesis is crucial for generating new perspectives of a scene and has attracted unprecedented attention over the past few years. NeRF (Mildenhall et al., 2020) is a groundbreaking paper introducing Neural Radiance Fields, a method for synthesizing novel views with impressive realism. Shortly after NeRF, numerous follow-up papers improved its quality (Barron et al., 2021; Hu et al., 2023), inference speed (Fridovich-Keil et al., 2022; Garbin et al., 2021), training efficiency (Sun et al., 2022a; Müller et al., 2022), and applicability (Meshry et al., 2019; Park et al., 2021a; Chan et al., 2022). The primary drawback of NeRF lies in its inefficiency. To address this limitation, a novel 3D representation called Gaussian Splatting (Kerbl et al., 2023) has emerged. Gaussian Splatting serves as a bridge between NeRF's high-quality view-dependent rendering and the hardware-friendly, efficient classical rasterization approach. Soon after its introduction, it quickly evolved and found uses in various improvements and applications, like dynamic scenes (Luiten et al., 2023; Yang et al., 2023d), generative models (Chen et al., 2023; Tang et al., 2023a), anti-aliasing (Yu et al., 2023), and relighting (Gao et al., 2023).

Simultaneous Localization and Mapping (SLAM). SLAM is a fundamental technology in robotics and computer vision, playing a crucial role in enabling machines to understand and navigate their surroundings. It involves the simultaneous process of creating a map of an unknown environment while determining the precise location of the observer within that environment. SLAM has applications ranging from autonomous vehicles and drones to augmented reality, contributing significantly to the development of intelligent systems capable of robustly interacting with and adapting to the world around them. ORB-SLAM2 (Mur-Artal and Tardós, 2017a), an open-source visual SLAM based on traditional computer vision techniques, is still a leading approach for localization accuracy. Over the past years, deep learning has contributed to many parts of the SLAM algorithms (Mokssit et al., 2023). With the advent of NeRF, the idea of exploiting implicit representation for SLAM was explored in iMAP (Sucar et al., 2021) and NICE-SLAM (Zhu et al., 2022). Such implicit representation can lead to the high-quality 3D reconstruction of the environment. Although these approaches are significantly slower than the traditional methods, they demonstrate promising quality in

3D reconstruction and opened a new line of research in the SLAM area.

This thesis encompasses contributions in the fields of depth estimation, novel view synthesis, and visual SLAM. Section 1.2 outlines our contributions in these domains in detail.

1.2 Thesis Outline and Contributions

The following three chapters in this thesis are based on three conference proceedings articles (Johari et al., 2021, 2022, 2023). In each chapter, we investigate a 3D computer vision task independently, and the chapters can be read in any order. An extensive supplementary material is also presented at the end of each chapter to make them self-contained. Nevertheless, the flow of the study unfolds as outlined below.

In our initial investigation, we focused on a specific challenge related to 3D understanding exclusively within the camera frustum. This undertaking aligns with the established domain of monocular depth estimation within the research framework of 3D computer vision. In **Chapter 2**, which is based on the work by Johari et al. (2021), we introduce **DepthInSpace**, a self-supervised deep-learning approach designed for depth estimation through structured-light cameras, addressing the growing demand for embedded depth sensors in contemporary smartphones. With the advent of structured-light cameras, depth sensing has become feasible with basic algorithms implementable on devices with computational constraints in real time. While traditional methods like block matching and semi-global matching have been employed by devices like Kinect V1 (Martinez and Stiefelhagen, 2013) and Intel RealSense (Keselman et al., 2017), learning-based approaches in this domain are limited.

The proposed DepthInSpace method leverages optical flow estimates derived from ambient information across multiple video frames to guide the training of a single-frame depth estimation network. This innovative approach aids in preserving edges and mitigating over-smoothing challenges, making depth estimation more effective. Additionally, we propose a technique to fuse data from multiple video frames, enhancing the accuracy of depth maps and minimizing artifacts, particularly in occluded areas. In another study, we demonstrate the efficacy of using the estimated fused depth maps as a self-supervision signal to refine a single-frame depth estimation network, resulting in an overall improvement in performance.

The models are thoroughly evaluated and compared with state-of-the-art counterparts using synthetic and newly introduced real datasets. With the lack of large-scale, precise ground-truth data, our end-to-end training of a deep neural network in a self-supervised manner becomes crucial. The implementation code, training procedure, and datasets are publicly available at https://www.idiap.ch/paper/depthinspace, facilitating further exploration and development in the field of self-supervised depth estimation.

In our subsequent investigation, we extended our focus beyond the camera frustum, delving into the realm of 3D sensing using multi-view data obtained either from a single object or densely sampled views from a limited camera trajectory of a real-world scene. This research aligns with the well-recognized area of novel view synthesis. **Chapter 3**, which is based on the work by Johari et al. (2022), introduces **GeoNeRF**, a novel and generalizable approach to photorealistic novel view synthesis leveraging neural radiance fields. Comprising two main stages, our method incorporates a geometry reasoner and a renderer. The geometry reasoner initiates the process by constructing cascaded cost volumes for nearby source views, facilitating sophisticated occlusion reasoning through a Transformer-based attention mechanism. Utilizing these cost volumes, the renderer employs classical volume rendering techniques to infer geometry and appearance, producing detailed images. Notably, this architecture excels in gathering information from consistent source views.

Our approach addresses a critical limitation of Neural Radiance Fields (NeRF) (Mildenhall et al., 2020), which requires scene-specific training, leading to time-consuming per-scene optimization with densely captured images. Recent approaches aiming to generalize NeRF to new scenes often fall short of understanding scene geometry and occlusions, resulting in undesirable artifacts. Building upon the foundation of MVSNeRF (Chen et al., 2021), we introduce key enhancements. The geometry reasoner utilizes cascaded cost volumes trained in a semi-supervised manner to obtain fine and high-resolution priors for conditioning the renderer. The renderer combines an attention-based model, handling information from diverse source views, with an auto-encoder network that aggregates information along a ray. Additionally, our method efficiently detects and excludes occluded views for each point in space, further enhancing rendering quality.

Moreover, GeoNeRF allows straightforward fine-tuning for a single scene, yielding competitive results compared to per-scene optimized neural rendering methods with significantly reduced computational costs. Experimental evaluations demonstrate GeoNeRF's superior performance over state-of-the-art generalizable neural rendering models across synthetic and real datasets. Lastly, as an extension, we propose an alternate model adaptable to RGBD images, leveraging depth information commonly available from depth sensors.

In summary, GeoNeRF advances the field of novel view synthesis by addressing the limitations of existing methods, offering a robust and efficient solution for generalizable neural rendering. The implementation source code and visualization videos are accessible at https://www.idiap.ch/paper/geonerf.

In our last contribution, we investigated a challenge involving the unrestricted movement of a camera within a real-world setting. This investigation is situated within the established and conventional domain of Simultaneous Localization and Mapping (SLAM) within the field of 3D computer vision research. **Chapter 4**, which is based on the work by Johari et al. (2023), introduces **ESLAM**, a novel and efficient method that adapts implicit neural representation to SLAM systems. ESLAM processes RGB-D frames sequentially, gradually reconstructing the scene while simultaneously estimating the current camera position. Traditional SLAM systems primarily focus on localization accuracy, while recent learning-based dense visual

SLAM methods provide global 3D maps with reasonable but limited reconstruction accuracy. Inspired by NeRF's capacity to reason about large-scale scene geometry, ESLAM builds upon NeRF-based dense SLAM methods like iMAP (Sucar et al., 2021) and NICE-SLAM (Zhu et al., 2022). Unlike iMAP's single MLP for geometry representation or NICE-SLAM's voxel grid storage for occupancies, ESLAM employs multi-scale axis-aligned feature planes to learn implicit Truncated Signed Distance Field (TSDF). This choice of 3D representation in ESLAM converges faster, delivers higher-quality reconstruction, and reduces the memory footprint growth rate.

Benchmarking on three challenging datasets validates ESLAM's superior performance, showcasing over 50% improvements in 3D reconstruction and camera localization accuracy compared to state-of-the-art methods. ESLAM achieves this while running up to $\times 10$ faster and without the need for pre-training. The method's inherent smoothness, attributed to the representation with feature planes, produces high-quality smooth surfaces without explicit smoothness loss functions. The findings position ESLAM as an efficient and accurate dense visual SLAM method with broad applicability. The implementation source code and visualization videos are accessible at: https://www.idiap.ch/paper/eslam

Finally, in **Chapter 5**, we summarize our findings, propose new directions for future research, and review recent advances in concurrent and successor studies.

Contributions Part II

2 DepthInSpace: Depth Estimation with Structured-Light Sensors

Disclaimer: This chapter is adapted from the following article – with permission of all co-authors and the conference:

Johari, M. M., Carta, C., and Fleuret, F. (2021). DepthInSpace: Exploitation and Fusion of Multiple Video Frames for Structured-Light Depth Estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6039-6048.

2.1 Abstract

We present DepthInSpace, a self-supervised deep-learning method for depth estimation using a structured-light camera. The design of this method is motivated by the commercial use case of embedded depth sensors in nowadays smartphones. We first propose to use the estimated optical flow from ambient information of multiple video frames as a complementary guide for training a single-frame depth estimation network, helping to preserve edges and reduce over-smoothing issues. Utilizing optical flow, we also propose to fuse the data of multiple video frames to get a more accurate depth map. In particular, fused depth maps are more robust in occluded areas and incur less in flying pixel artifacts. We finally demonstrate that these more precise fused depth maps can be used as self-supervision for fine-tuning a single-frame depth estimation network to improve its performance. Our models' effectiveness is evaluated and compared with state-of-the-art models on both synthetic and our newly introduced real datasets. The implementation source code, training recipe, and both synthetic and captured real datasets are available in the following link: https://www.idiap.ch/paper/depthinspace.

2.2 Introduction

With the advent of structured-light cameras, depth-sensing became conceivable with basic algorithms implementable on devices with computational constraints in real time. For instance, Kinect V1 uses a correlation-based block matching technique (Scharstein and Szeliski, 1920), and Intel RealSense (Keselman et al., 2017) employs a semi-global matching scheme (Hirschmuller, 2007). However, learning-based approaches in this field are relatively limited. Ryan Fanello et al. (2017) propose a computationally efficient feature-matching method. Projecting image patches to compact binary representation is proposed in UltraStereo (Fanello et al., 2017) to achieve a low-complex matching scheme. HyperDepth (Ryan Fanello et al., 2016) casts the problem of depth estimation as a classification-regression task, which it solves using an ensemble of cascaded random forests. However, HyperDepth assumes the availability of ground-truth labels either from high-accuracy sensors or exhaustive stereo-matching search algorithms.

Due to the lack of large-scale, precise ground-truth data, end-to-end training of a deep neural network in a self-supervised manner has been at the center of attention recently. ActiveStereoNet (Zhang et al., 2018b) uses Siamese networks for predicting disparity and proposes a novel photometric loss function based on a Local Contrast Normalization (LCN) scheme for training. A separate color sensor is used in the research by Kleitsiotis et al. (2019) to enhance the performance of the approach by Zhang et al. (2018b). Riegler et al. (2019) exploit the photometric loss function of Zhang et al. (2018b) and propose an edge-detection network along with an edge-aware smoothness loss function to overcome the issue of edge fattening. They also introduce another loss function that leverages the information of other video frames to supervise the disparity estimation network's training. To do so, they use the estimated disparity and camera pose parameters to transform pixels into a 3D point cloud and apply the consistency of the predicted depth of matched pixels across multiple frames.

We take the work by Riegler et al. (2019) as the baseline, and our contributions in this chapter are as follows:

- We propose a novel training scheme that uses optical flow predictions from ambient images to find matched pixels independently of the estimated disparities, which stabilizes the training and enhances accuracy. Our sensor can capture ambient images conveniently, and we exploit this feature in this regard.
- We extend this model to fuse information from multiple video frames to obtain more precise disparity maps with sharper edges and fewer artifacts.
- We finally propose to exploit the resulting fused disparity maps to fine-tune a single-frame disparity estimation network.

2.3 Related Work

Active Depth Estimation: The setup usually consists of a camera and a projector which projects a random but known pattern of dots into the scene. Depending on the depth of objects in the environment, the camera receives a deformed shape of the projected pattern, and this phenomenon could be used in depth estimation algorithms. Such algorithms include basic searching for correspondences in Kinect V1 (Martinez and Stiefelhagen, 2013), computationally efficient learning-based techniques (Fanello et al., 2017; Ryan Fanello et al., 2016; Chen and Koltun, 2014), and a deep neural network trained end-to-end to estimate disparity map directly (Zhang et al., 2018b; Kleitsiotis et al., 2019; Riegler et al., 2019).

Leveraging Multiple Frames: Utilizing multiple frames for depth estimation includes but is not limited to structured-light sensors (Riegler et al., 2019). In the studies by Godard et al. (2017), Xie et al. (2016), and Kuznietsov et al. (2017), the second image of a stereo camera is regarded as another video frame. Explicit utilization of multiple video frames of a conventional camera for self-supervision is proposed in numerous researches (Zhan et al., 2018; Zhou et al., 2017; Bian et al., 2019; Godard et al., 2019; Guizilini et al., 2020; Pillai et al., 2019; Casser et al., 2019). Fusing the information of multiple frames during inference is employed in RGB depth estimation models like DeepV2D (Teed and Deng, 2019), DeepMVS (Huang et al., 2018), DeepSFM (Wei et al., 2020), and DPSNet (Im et al., 2018) in the form of aggregating volume cost representations. In these papers, the aggregation is done by simple pooling operations (DeepV2D and DeepMVS) or by performing convolution on the 2D grid (DeepSFM and DPSNet). Such approaches would fail in the context of structured-light images, where the projector also moves with the camera. As a result of the moving projector, the scene is textured with the projected dots differently, and the camera captures an entirely new scene in each frame. Simply warping frames together and aggregating them on the 2D grid will limit the performance since the dots' information is meaningless in the warped frames and interferes with the fusion process. We tackle this issue in Section 2.4.2, where we perform fusion and convolution in the continuous 3D space to leverage the consistency of geometry there maximally. Unfortunately, all the aforementioned models are designed to work with RGB images, and we cannot evaluate them for structured-light images through experiments. However, we examine how the aggregation of frames on the 2D grid would fail for these images in this study.

Optical Flow and Depth Estimation: Numerous pieces of research in passive depth estimation suggest taking advantage of consistency between optical flow prediction and camera ego-motion between consecutive video frames. Wang et al. (2019), Yin and Shi (2018), Zou et al. (2018), and Ranjan et al. (2019) claim that simultaneous training of an optical flow network and a depth estimation network can benefit both tasks and result in a better performance than training those individually. Notably, Luo et al. (2020) proposes a novel framework capable of fine-tuning a general monocular depth estimation network during test time by leveraging a pre-trained optical flow estimation network. Although it is not common in the context of active stereo depth sensing, there is adequate ambient information in captured images to exploit and predict optical flow between frames and improve the quality of depth estimation accordingly.

Convolution in Point Cloud: In the context of point cloud processing, some novel techniques are proposed that perform convolution on points in the continuous 3D space resembling convolutional neural networks of regular grid structures. Thomas et al. (2019), Li et al. (2018b), Xu et al. (2018), Wu et al. (2019), Boulch (2020), and Wang et al. (2018b) propose models that are capable of applying convolution on unstructured and unordered data and work well on point cloud benchmark tasks and datasets. For 2D grid-style data, when depth information is available, it is plausible to transform points into 3D space and leverage such continuous convolutions. Such an approach is presented by Chen et al. (2019), where the method jointly benefits from conventional 2D convolution and parametric continuous convolution introduced by Wang et al. (2018b).

2.4 Method

We build DepthInSpace (DIS) model upon the Connecting the Dots (CTD) model by Riegler et al. (2019). CTD suggests using two separate networks, one for estimating the disparity, and the other for detecting the edges in the images. The edge detector is weakly supervised with the ambient images, which are the same as dot images except that the projector is off during photo capture. Obtaining ambient data is considerably cheaper than ground-truth depth data; however, the edge detection network is proposed to reduce the number of ambient images required for training.

We claim ambient images contain more valuable information than only the objects' edges. The sensor that we use is equipped with a programmable switch that can capture both dot images and ambient images at no additional cost. Accordingly, we discard the edge detection network and replace the CTD's smoothing loss function with a loss that directly extracts edges from ambient images. Also, we predict the optical flow from ambient images to find the matched pixels and introduce a new loss which encourages geometric consistency between them. Our proposed loss replaces the geometric loss in CTD and is preferable in two regards. First, CTD uses the momentary predicted depth and ego-motion of the camera to find the matched pixels. As a result, the optimization landscape changes rapidly during training and could result in instability of training. Secondly, the error in momentary predicted depths participates in the procedure of finding matched pixels and leads to degraded performance. In addition, the matching scheme with optical flow provides more flexibility to detect mistakenly matched pixels and exclude them from contributing to the loss function. We use LiteFlowNet (Hui et al., 2018) pre-trained on MPI Sintel (Butler et al., 2012) for optical flow, which is a lightweight and fast model, but it has comparable performance to computational and memory resource expensive models like FlowNet2 (Ilg et al., 2017).

2.4.1 Single-Frame Disparity Estimation

Our DepthInSpace Single-Frame (DIS-SF) model takes the CTD model (Riegler et al., 2019) as a baseline and modifies two of its loss functions: we incorporate a novel multi-view loss

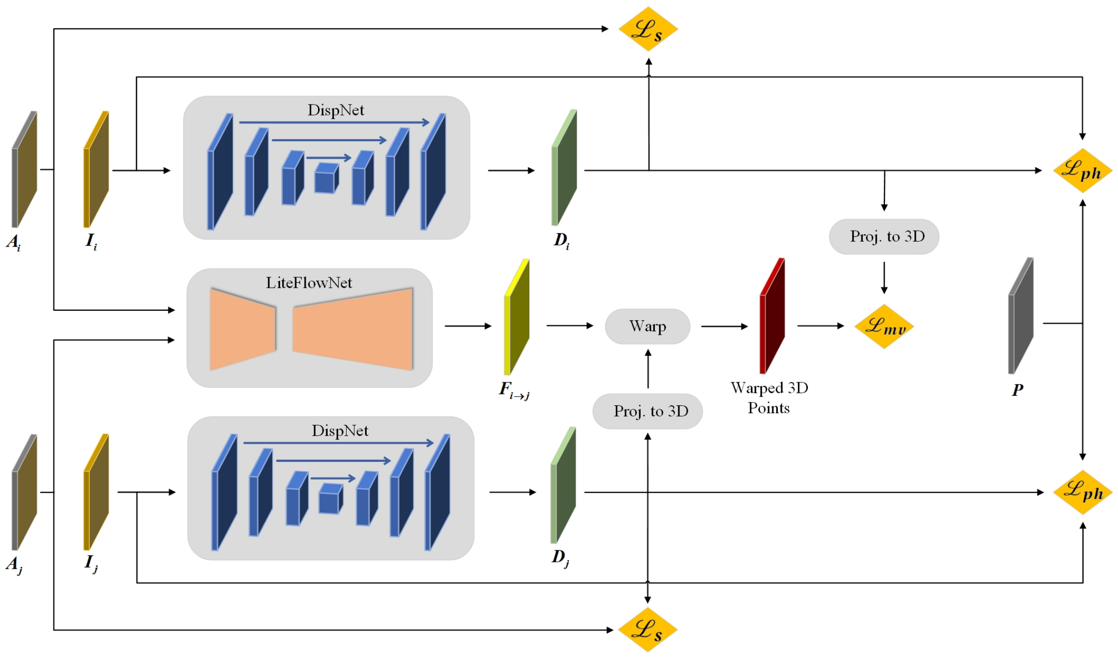


Figure 2.1: The training scheme of our DIS-SF model for a sample pair of frames i and j, and a reference pattern $\boldsymbol{P}$. The dot images $\boldsymbol{I_i}$ and $\boldsymbol{I_j}$ are fed to the DispNet (Mayer et al., 2016) separately to predict disparities $\boldsymbol{D_i}$ and $\boldsymbol{D_j}$. On another path, LiteFlowNet (Hui et al., 2018) generates optical flow of these two frames $\boldsymbol{F_{i\to j}}$ exploiting ambient images $\boldsymbol{A_i}$ and $\boldsymbol{A_j}$ jointly. The photometric loss $\mathscr{L}_{ph}$ and the smoothness loss $\mathscr{L}_s$ are applied to images separately, whereas the multi-view loss $\mathscr{L}_{mv}$, which imposes consistency of predicted depths between two frames, is applied pairwise (see Section 2.5). This scheme is employed for every pair of images from the same scene. The block **Warp** denotes bilinear 2D warping via optical flow and the block **Proj. to 3D** means projecting points into 3D space using the disparities and the camera's intrinsic parameters and adjusting the view angle of points using the camera's extrinsic parameters. After training and for disparity inference, DispNet (Mayer et al., 2016) takes a single dot image $\boldsymbol{I}$ and estimates a disparity map $\boldsymbol{D}$ as output.

function leveraging optical flow predictions and an improved edge-aware smoothness loss. The training scheme of our DIS-SF model is presented in Figure 2.1. The photometric loss $\mathscr{L}_{ph}$ enforces consistency between the input image and the warped reference pattern via the estimated disparity map. For smoothness loss $\mathscr{L}_s$, we propose using an edge-aware one similarly to Godard et al. (2017), Godard et al. (2019), and Pillai et al. (2019), except that we extract the edge information directly from the ambient images.

Furthermore, we introduce a novel multi-view loss $\mathscr{L}_{mv}$, which enforces the consistency of the estimated depths between two different views with the help of bilinear warping via optical flow predictions. Note that the photometric loss and smoothness loss apply to each image individually, whereas the multi-view loss applies to all possible permutations of image pairs from the same scene. For more details about the loss functions, refer to Section 2.5.

We use DispNet (Mayer et al., 2016) for inferring disparity. We also apply Local Contrast Normalization (LCN) preprocessing, suggested by Zhang et al. (2018b) and Riegler et al. (2019), to both dot images $\boldsymbol{I}$ and the reference pattern $\boldsymbol{P}$. Although we use ambient images $\boldsymbol{A}$ in our training scheme, we do not directly employ them as DispNet's input. This makes data preparation more convenient during inference, and DispNet (Mayer et al., 2016) predicts disparity maps $\boldsymbol{D}$ only based on dot images $\boldsymbol{I}$. Instead, the pairs of ambient images are exploited as the input of LiteFlowNet (Hui et al., 2018) to predict the optical flow map $\boldsymbol{F}$. More discussion on how we use pre-trained LiteFlowNet with ambient images, while it is designed to work with RGB images, as well as an ablation study are provided in the supplementary material in Sections 2.8.3 and 2.8.5.

2.4.2 Multi-Frame Disparity Estimation

Our Multi-Frame (DIS-MF) model combines the information of other frames from the same scene into one frame and generates more accurate disparities. We assume an initial imperfect disparity map is available for each frame beforehand, and we attempt to increase the quality of the disparities by fusing the frames. In this regard, we take the outputs of our DIS-SF model as the imperfect disparities. Compared to traditional RGB depth estimation, aggregating data of multiple frames is more efficacious in a structured-light setup because the performance of depth sensing depends on how the dots touch the objects in the environment. Thus, the data contained in the frames are less correlated.

Let $\boldsymbol{\phi} \in \mathbb{R}^{C\times H\times W}$ denote a feature map of size $H \times W$ with C channels, and $\boldsymbol{X} \in \mathbb{R}^{3\times H\times W}$ denote the corresponding 3D points obtained using the imperfect disparities and camera projection matrix $\boldsymbol{K} \in \mathbb{R}^{3\times 3}$. Let us assume we have a pair of images with feature maps of $(\boldsymbol{\phi_i}, \boldsymbol{\phi_j})$ and 3D points of $(\boldsymbol{X_i}, \boldsymbol{X_j})$. Frame i is assumed as the target frame, and we want to fuse the information of $\boldsymbol{\phi_j}$ into $\boldsymbol{\phi_i}$. Our model's first step is warping both feature map $\boldsymbol{\phi}$ and 3D points $\boldsymbol{X}$ on the 2D grid via optical flow predictions $\boldsymbol{F_{i\to j}}$ and $\boldsymbol{F_{j\to i}}$. Optical flow warping places the data of the frames on the 2D grid such that corresponding data of the frames appear in each other's neighborhood on the 2D grid.

Let $\boldsymbol{\phi_{j\to i}} = w^{j\to i}(\boldsymbol{\phi_j})$ and $\boldsymbol{X_{j\to i}} = w^{j\to i}(\boldsymbol{X_j})$ denote warped features and warped points, where $w^{j\to i}(\cdot)$ stands for bilinear 2D warping via the optical flow $\boldsymbol{F_{i\to j}}$. We also define a binary mask map $\boldsymbol{M_{j\to i}} \in \{0,1\}^{1\times H\times W}$ which indicates if the warped data is valid and should be allowed to participate in our fusion framework. We construct $\boldsymbol{M_{j\to i}}$ by evaluating the forward-backward consistency of optical flow predictions, similarly to Zou et al. (2018) and Meister et al. (2018):

$$\boldsymbol{M_{j\to i}} = |\boldsymbol{F_{i\to j}} + w^{j\to i}(\boldsymbol{F_{j\to i}})|^2 < 0.01 \times (|\boldsymbol{F_{i\to j}}|^2 + |w^{j\to i}(\boldsymbol{F_{j\to i}})|^2) + 0.5 \tag{2.1}$$

Despite having all warped data and their validation mask map on the same 2D grid, we do not perform fusion naively on the grid space. As we already mentioned in Section 2.3, warped features in the structured-light setup contain interfering data of warped dots that make the fusion task complicated. Instead, we propose a fusion block that performs fusion

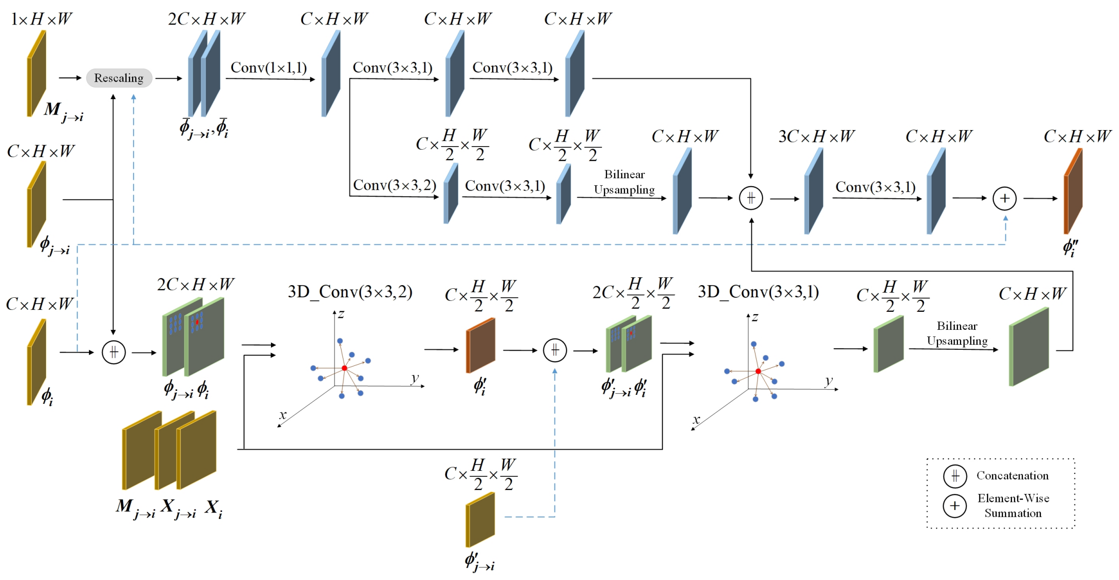


Figure 2.2: Internal architecture of our proposed fusion block, whose details of utilization in our DIS-MF model are illustrated in Section 2.4.2 and Figure 2.3. We depict how features of an auxiliary frame $\boldsymbol{\phi}_{\boldsymbol{j}\to\boldsymbol{i}}$ are being fused into the target frame's features $\boldsymbol{\phi}_{\boldsymbol{i}}$. Binary mask map $\boldsymbol{M}_{\boldsymbol{j}\to\boldsymbol{i}}$, 3D points of the target frame $\boldsymbol{X}_{\boldsymbol{i}}$ and warped frame $\boldsymbol{X}_{\boldsymbol{j}\to\boldsymbol{i}}$, and the warped result of the first 3D convolution of the auxiliary frame $\boldsymbol{\phi}'_{\boldsymbol{j}\to\boldsymbol{i}}$ are also inputs of this block. $\boldsymbol{\phi}''_{\boldsymbol{i}}$ stands for the output of this block, and $\boldsymbol{\phi}'_{\boldsymbol{i}}$ represents the output of the first 3D convolution required for fusing into other frames' fusion blocks. Conv$(k\times k,s)$ and 3D_Conv$(k\times k,s)$ denote 2D and continuous 3D convolution respectively, with kernel size of k and stride s, and the block **Rescaling** denotes the operations described in Equation (2.4).

and convolution in the continuous 3D space. Our fusion block also has a sense of faulty imperfect disparities and can prevent those points from contributing to the aggregation. The details of our fusion block and its utilization in our DIS-MF network architecture are as follows.

Fusion Block: Chen et al. (2019) suggest when depth information of a 2D image is available, it is conceivable to exploit continuous convolution in the 3D space and benefit from both 2D and 3D data processing simultaneously. Such a proposal is consistent with the idea of merging the data of multiple frames as the projected points in the 3D space could be processed regardless of their camera pose. Inspired by them, we propose a fusion block capable of fusing several feature maps originating from different frames into the target frame's feature map. For the sake of simplicity, let us assume we only have two frames and intend to merge the feature map $\boldsymbol{\phi}_{\boldsymbol{j}}$ into the target feature map $\boldsymbol{\phi}_{\boldsymbol{i}}$. The functionality of the fusion block is illustrated in Figure 2.2. We use the continuous 3D convolution (Wang et al., 2018b) as the core element of our fusion block. Most architectures that exploit 3D convolution on the point cloud require running exhaustive search algorithms to find points in the neighborhood (Chen et al., 2019; Li et al., 2018b; Xu et al., 2018; Wu et al., 2019; Boulch, 2020; Wang et al., 2018b), which is infeasible to perform on dense data such as ours. For instance, Chen et al. (2019) pre-compute the indices of nearest neighbors for all points. To mitigate the issue, we propose a novel technique that is practical in real-time processing. Since our data is not fully unstructured, we suspect points that are close in 3D space will be close on the 2D grid map if they are warped to the same camera perspective, but not vice versa.

Accordingly, we form the concatenated feature map $[\boldsymbol{\phi}_{\boldsymbol{j \to i}}, \boldsymbol{\phi}_{\boldsymbol{i}}]$ and point map $[\boldsymbol{X}_{\boldsymbol{j \to i}}, \boldsymbol{X}_{\boldsymbol{i}}]$ and slide a 3×3 window over each 2D grid map simultaneously and perform convolution only on points inside the sliding window similarly to a conventional CNN. The difference is, that instead of performing a weighted sum with learnable parameters, we search for the nearest points and perform continuous convolution. For simplifying the equations, let $\boldsymbol{\phi}_{\boldsymbol{i \to i}} = \boldsymbol{\phi}_{\boldsymbol{i}}$, $\boldsymbol{X}_{\boldsymbol{i \to i}} = \boldsymbol{X}_{\boldsymbol{i}}$, and $\boldsymbol{M}_{\boldsymbol{i \to i}} = \vec{\mathbf{1}}$. Also, let $\boldsymbol{\phi}(h, w)$ and $\boldsymbol{X}(h, w)$ represent the features and the coordinate of the position (h, w) on the grid map where $0 \le h < H$ and $0 \le w < W$. We first search for the nearest points to the center point of the sliding window on the target frame i:

$$l^*(h, w), m^*(h, w), n^*(h, w) = k\text{-}\underset{\substack{l \in \{i,j\} \\ -1 \le m \le +1 \\ -1 \le n \le +1}}{\arg\min} \frac{\left| \boldsymbol{X}_{\boldsymbol{l \to i}}(h+m, w+n) - \boldsymbol{X}_{\boldsymbol{i}}(h, w) \right|}{\boldsymbol{M}_{\boldsymbol{l \to i}} + \epsilon} \tag{2.2}$$

where k-$\arg\min g(\cdot)$ returns the k indices that minimize the function $g(\cdot)$, and ϵ is a small constant. $\boldsymbol{M}_{\boldsymbol{l \to i}}$ is used in the denominator to exclude invalid points, and we set $k = 9$ to ensure all returned indices correspond to valid pixels due to the window size 3×3. To extend the model to fuse more than two frames, l in Equation (2.2) should span all available frames rather than only $\{i, j\}$. The convolution's result is:

$$\boldsymbol{\phi}'_{\boldsymbol{i}}(h, w) = \Psi \times \sum_{l^*, m^*, n^*} \Big(\boldsymbol{\phi}_{\boldsymbol{l^* \to i}}(h+m^*, w+n^*) \odot \text{MLP}\big(\boldsymbol{X}_{\boldsymbol{l^* \to i}}(h+m^*, w+n^*) - \boldsymbol{X}_{\boldsymbol{i}}(h, w) \big) \Big) \tag{2.3}$$

where MLP is a multi-layer perceptron mapping 3D vectors to C-dimensional weights, $\odot$ denotes element-wise product, and Ψ is a $C \times C$ learnable weight matrix. This implementation can be regarded as a continuous version of separable convolution. The MLP and weighted sum perform depth-wise convolution, while the linear transformation resembles 1×1 convolution (Chen et al., 2019).

As shown in Figure 2.2, we adopt two 3D convolutions in each fusion block. Accordingly, we warp the other frames' outputs of the first 3D convolution to the target frame $\boldsymbol{\phi}'_{\boldsymbol{j \to i}}$ and fuse them into the second 3D convolution as well. We also employ traditional 2D CNNs in the fusion block because there are some shortcomings to 3D convolution, such as edge fattening near the boundaries of objects and background. To merge the feature maps in 2D CNNs, we handle invalid points differently by proposing a scheme similar to dropout (Srivastava et al., 2014). To do so, we first zero out features of invalid points, and then rescale the remaining valid features inversely proportionally to the number of valid frames for each point on the 2D

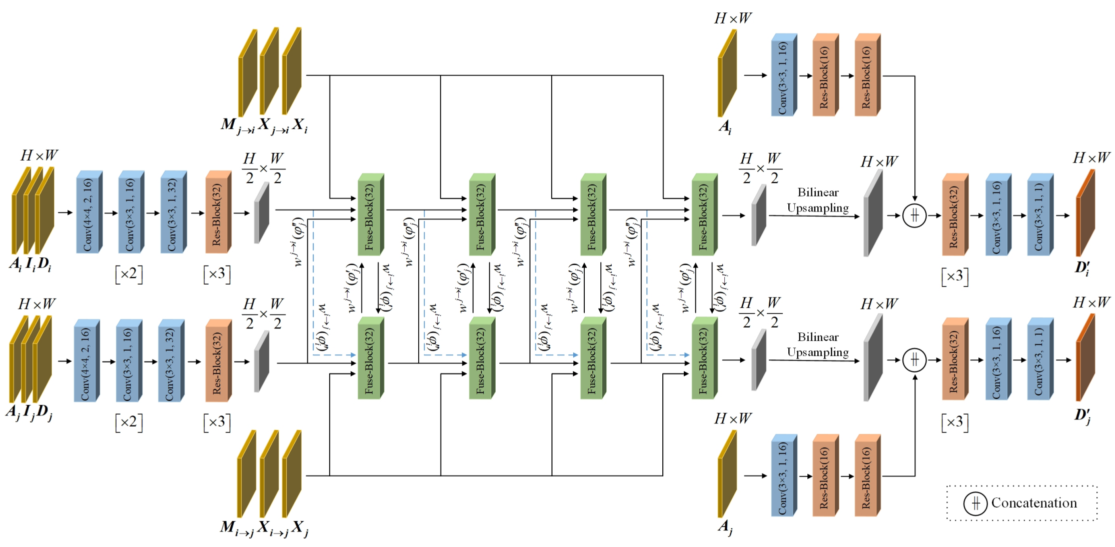


Figure 2.3: Our DIS-MF network architecture when only two frames i and j are combined. Warping the first 3D convolution output $\boldsymbol{\phi}'$ and the final output of each fusion block $\boldsymbol{\phi}''$ using the relative optical flows are denoted by $w^{i\to j}(\cdot)$ and $w^{j\to i}(\cdot)$. Note that $\boldsymbol{D}$ stands for imperfect disparity participating as one of the inputs, and $\boldsymbol{D}'$ represents the final predicted disparity of the model. This figure depicts the inference network of our DIS-MF model. For training the DIS-MF model, this network replaces those individual DispNet (Mayer et al., 2016) networks in the DIS-SF model in Figure 2.1, and the same scheme and loss functions (see Section 2.5) are adopted.

grid. Specifically:

$$\forall l \in \{i,j\}: \bar{\boldsymbol{\phi}}_{\boldsymbol{l\to i}} = \frac{\boldsymbol{\phi}_{\boldsymbol{l\to i}} \times \boldsymbol{M}_{\boldsymbol{l\to i}}}{\sum\limits_{p\in\{i,j\}} \boldsymbol{M}_{\boldsymbol{p\to i}}} \tag{2.4}$$

The 3D convolutions along with 2D CNNs jointly construct the fusion block, which is capable of processing high-resolution feature maps and effectively benefits from the information of other frames from the same scene. SELU nonlinearity (Klambauer et al., 2017) and Group Norm (Wu and He, 2018) are used after each convolution. We prefer Group Norm to Batch Norm (Ioffe and Szegedy, 2015) in our model because Group Norm statistics are independent of the number of samples in a batch and make training large networks feasible with smaller batch sizes.

Network Architecture: Figure 2.3 illustrates the network architecture of our DIS-MF model. The architecture includes three sections as follows. The preprocessing section takes the images ($\boldsymbol{I}, \boldsymbol{A}$) and the imperfect disparity $\boldsymbol{D}$ as input and generates high-level feature maps for each frame individually. Next, the feature maps are fed into a cascaded series of fusion blocks, along with their corresponding 3D points $\boldsymbol{X}$ and binary masks $\boldsymbol{M}$ required for merging and 3D convolutions to obtain fused feature maps. Warping with the optical flow is employed whenever any data on a 2D grid map needs to be warped to another frame's 2D grid.

Lastly, the fused feature maps go through a refinement structure to preserve high-resolution details such as edges and reduce distortions resulting from combining frames. Our refinement section is inspired by the one by Zhang et al. (2018b) but takes the upsampled fused features and the ambient image as inputs. In both the preprocessing and refinement sections, we exploited residual blocks introduced by He et al. (2016) to promote gradient backpropagation and expedite the training process.

An ablation study of design choices for the DIS-MF network architecture is provided in the supplementary material in Section 2.8.5.

2.4.3 Fine-Tuning the Single-Frame Model

For purposes where resources are limited during inference, we propose an alternative approach to exploit the scheme of fusing image frames. We suggest that after training the DIS-MF model, the produced disparities can be used as an auxiliary loss function to supervise and fine-tune the single-frame network. The resulting model, DepthInSpace Fine-Tuned Single-Frame (DIS-FTSF), can yield more accurate disparity maps with no additional memory or computation cost during inference compared with DIS-SF.

2.5 Loss Functions

Here we introduce the loss functions employed in our models. Let $\Gamma = \{\boldsymbol{I_i}, \boldsymbol{A_i}\}_{i=0}^{N-1}$ denote the image samples from the same scene. The overall loss function consists of a photometric loss $\mathscr{L}_{\boldsymbol{ph}}$, a smoothness loss $\mathscr{L}_{\boldsymbol{s}}$, a multi-view loss $\mathscr{L}_{\boldsymbol{mv}}$, and a pseudo-ground truth loss $\mathscr{L}_{\boldsymbol{pgt}}$:

$$\mathscr{L} = \frac{1}{N}\sum_{i\in\Gamma}(\mathscr{L}^{\boldsymbol{i}}_{\boldsymbol{ph}} + \lambda_1\mathscr{L}^{\boldsymbol{i}}_{\boldsymbol{s}} + \lambda_2\mathscr{L}^{\boldsymbol{i}}_{\boldsymbol{pgt}}) + \frac{1}{N(N-1)}\sum_{i,j\in\Gamma}\lambda_3\mathscr{L}^{\boldsymbol{ij}}_{\boldsymbol{mv}} \tag{2.5}$$

where $\{\lambda_k\}_{k=1}^{3}$ are weighting constants, which do not necessarily take the same value in all of our models.

Let $\boldsymbol{D}$ denote the disparity map, $\tilde{\boldsymbol{I}}$ denote the local contrast normalized input image, and $\boldsymbol{P}$ denote the local contrast normalized reference dot pattern. Similarly to CTD, we employ the smooth Census transform (Hafner et al., 2013), represented by $\| \cdot \|_C$, in our photometric loss:

$$\mathscr{L}^{\boldsymbol{i}}_{\boldsymbol{ph}} = \sum_{h,w} \| \tilde{\boldsymbol{I}}_{\boldsymbol{i}}(h,w) - \boldsymbol{P}\big(h, w - \boldsymbol{D_i}(h,w)\big) \|_C \tag{2.6}$$

Since we assume the availability of ambient images, we introduce an edge-aware smoothness loss similar to the ones by Godard et al. (2017) and Godard et al. (2019). The smoothness loss imposes consistency between disparity map discontinuities and edges in the ambient image:

$$\mathscr{L}^{\boldsymbol{i}}_{\boldsymbol{s}} = |\nabla_h \boldsymbol{D_i}| e^{-\beta|\nabla_h \boldsymbol{A_i}|} + |\nabla_w \boldsymbol{D_i}| e^{-\beta|\nabla_w \boldsymbol{A_i}|} \tag{2.7}$$

where ∇_h and ∇_w stand for 2D spatial gradients and β is a constant. Moreover, we impose the consistency between the predicted depths in each pair of images from the same scene. Let $\boldsymbol{X_i}$ and $\boldsymbol{X_j}$ denote the 3D point clouds of the two frames obtained using the momentary predicted disparities and camera intrinsic matrix. Our multi-view loss is:

$$\mathscr{L}_{\boldsymbol{mv}}^{\boldsymbol{ij}} = \left| \left\langle \boldsymbol{X_i} - w^{j\to i}\left(\boldsymbol{T_{j\to i}} \times [\boldsymbol{X_j}, \vec{\mathbf{1}}]\right) \right\rangle_z \right| \times \boldsymbol{M'_{j\to i}} \tag{2.8}$$

where $\boldsymbol{T_{j\to i}} \in \mathbb{R}^{3\times 4}$ is the transformation matrix consisting of ego motion parameters, $\vec{\mathbf{1}}$ is an all one matrix, and $\langle \cdot \rangle_z$ operator returns the depth z of its input 3D vector. $\boldsymbol{M'_{j\to i}}$ is a binary mask map validating warped points similarly to $\boldsymbol{M_{j\to i}}$ in Section 2.4.2, but it strictly excludes low confidence points from supervising the training. For more details regarding $\boldsymbol{M'_{j\to i}}$, refer to the supplementary material in Section 2.8.2.

Lastly, only in our DIS-FTSF model, we use the more accurate fused disparity $\boldsymbol{D'}$ as pseudo-ground truth to improve the quality of the imperfect disparity $\boldsymbol{D}$. We impose the L1 consistency between $\boldsymbol{D}$ and $\boldsymbol{D'}$ as an auxiliary loss:

$$\mathscr{L}_{\boldsymbol{pgt}}^{\boldsymbol{i}} = |\boldsymbol{D_i} - \boldsymbol{D'_i}| \tag{2.9}$$

2.6 Experiments

Datasets: To evaluate our models and compare them with existing methods, we examine the accuracy of depth estimation on three synthetic datasets and one real dataset. We used the tool provided by CTD (Riegler et al., 2019) to render the synthetic data. Rendering is done in the same experimental setup as CTD with the same objects of the ShapeNet Core dataset (Chang et al., 2015), but the images are captured by a sensor whose parameters are set similarly to our own hardware. One dataset is rendered using the Kinect dot pattern for projection, and the second dataset is generated utilizing our own theoretical dot pattern for the projector. For the last synthetic dataset, we projected and captured the dot pattern in a real laboratory environment and used the observed pattern for rendering the dataset. In this regard, we use a virtual projector with the same parameters as the capturing camera.

We incorporated multiple datasets because different dot patterns could lead to different depth-sensing performances. The denser the dots are, the better the performance is. However, choosing a dot pattern could be restricted by hardware limitations or available illumination power. That is why we examine the models' performances over different projected dot patterns. For each synthetic dataset, we create 8192 sequences for training, 512 sequences for validation, and 512 sequences for testing. Each sequence contains 4 pairs of dot images and ambient images from the same scene.

We also evaluate the models on a smaller real dataset to show the generalization of our method in an actual setup. The data include 148 sequences of 4 pairs of dot images and ambient

Dataset	Method	$o(0.5)$	$o(1)$	$o(2)$	$o(5)$
Synthetic (Kinect Pattern)	SGM	10.36	9.13	8.76	2.45
	HyperDepth[a]	4.38	3.22	2.69	2.39
	CTD	2.74	1.45	0.77	0.24
	DIS-SF	2.11	1.13	0.59	0.16
	DIS-FTSF	**1.92**	**1.00**	**0.51**	**0.14**
	DIS-MF	1.59	0.72	0.33	0.10
Synthetic (Our Pattern)	SGM	12.93	11.64	11.22	4.06
	HyperDepth[a]	7.35	6.48	6.11	5.86
	CTD	3.38	1.71	0.85	0.28
	DIS-SF	2.31	1.24	0.62	0.19
	DIS-FTSF	**1.96**	**0.95**	**0.45**	**0.12**
	DIS-MF	1.58	0.71	0.32	0.10
Synthetic (Observed Pattern)	SGM	12.45	10.37	9.55	4.83
	HyperDepth[a]	6.13	4.92	4.34	4.00
	CTD	3.76	2.25	1.03	0.37
	DIS-SF	3.66	2.16	1.00	0.23
	DIS-FTSF	**2.87**	**1.48**	**0.66**	**0.17**
	DIS-MF	2.46	1.24	0.54	0.14
Real	SGM[b]	25.54	19.23	17.75	16.96
	HyperDepth[a]	34.62	25.09	22.49	21.77
	CTD	22.74	9.26	3.79	**1.00**
	DIS-SF	17.95	7.93	3.59	1.14
	DIS-FTSF	**17.06**	**7.48**	**3.47**	1.11
	DIS-MF	16.07	7.14	3.41	1.09

[a] HyperDepth is a supervised model trained with ground truth.
[b] We evaluated all models on the full image. SGM performs poorly on real data due to large disparities in the dataset and its incapability of predicting valid depths on a large portion of the image (whereas learning models extrapolate in those areas). As an example, if we evaluated models on a cropped area of the depth maps, $o(0.5)$ and $o(1)$ would drop to 15.56 and 8.81 for SGM, and 13.06 and 5.08 for DIS-FTSF.

Table 2.1: Quantitative comparison of the SGM algorithm (Hirschmuller, 2007), HyperDepth (Ryan Fanello et al., 2016), and CTD (Riegler et al., 2019) versus our DIS-SF, DIS-FTSF, and DIS-MF models. Numbers are percentages of outliers $o(t)$, that is the fraction of pixels for which the estimated disparity is more than t away from ground truth. We indicate in bold the best performance among single-frame methods (*i.e.*, all but our DIS-MF model, which, as expected, performs the best).

images captured from 4 different scenes. The sensor we use is equipped with a programmable switch, enabling the projector to be on and off, so it can capture dot images and ambient images alternately at the rate of 15 fps each. Given the capturing rate, each pair of a dot image and an ambient image captures the same scene approximately. We put aside 18 sequences

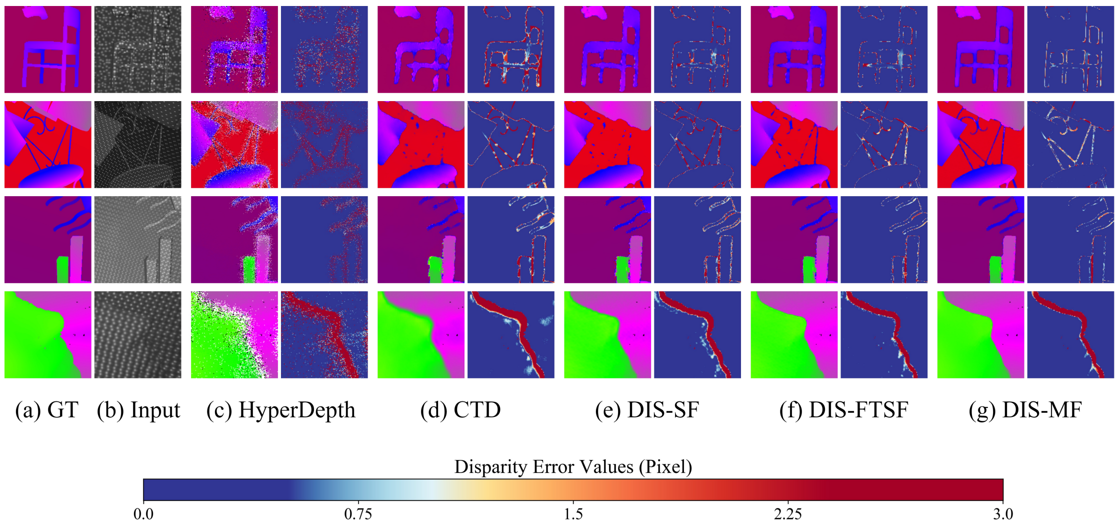


Figure 2.4: Qualitative results of the methods and their corresponding error maps. (a) Ground truth disparity map. (b) Input dot image with the projected pattern. (c) HyperDepth (Ryan Fanello et al., 2016). (d) CTD (Riegler et al., 2019). (e) Our DIS-SF model. (f) Our DIS-FTSF model. (g) Our DIS-MF Model. Each row represents a sample corresponding to each dataset in Table 2.1. Points for which the ground truth data is unavailable are excluded from the evaluation, and the color bar represents the disparity error map in pixels. For more sample images and extended qualitative evaluations, refer to the supplementary material in Section 2.8.6.

for validating and testing and utilized 130 sequences in training. To obtain accurate ground truth we used a 3D scanner, the data of which is only used for evaluation. Due to the scanner limitations, we take a set of partial scans that best cover the scene. These are fused together to create a 3D model using the point-to-plane variant of the ICP algorithm (Chen and Medioni, 1992). A 3D mesh is then produced using the Ball-Pivoting algorithm (Bernardini et al., 1999). For estimating the camera motion parameters, the same ICP variant is used to align the ground truth 3D model and the 3D model obtained from the structured-light sensor via the block matching technique.

More details of the datasets and also implementation of our models are provided in the supplementary material in Section 2.8.4.

Metrics: We use the percentage of outliers $o(t)$, which was used by Riegler et al. (2019) for quantitative evaluation, which is the percentage of pixels where the difference between the estimated and the ground truth disparities is greater than t.

Comparison with existing methods: We compare our models with Semi-Global Matching (SGM) algorithm (Hirschmuller, 2007), HyperDepth (Ryan Fanello et al., 2016), and CTD (Riegler et al., 2019). We observed through experiments that the window size of 13 for the SGM algorithm best suits our dataset. For HyperDepth, we used the same reimplementation code provided by Riegler et al. (2019) with the hyperparameters that yield the best results in the original paper (Ryan Fanello et al., 2016). Since HyperDepth is a supervised method, we used the ground truth depth maps for training this model.

When training either CTD or our models on the real dataset, we use the pre-trained weights obtained from the synthetic data in order to speed up the training process. Moreover, due to the limitations of the 3D scanner we used to capture ground truth, we had to put objects very close to the camera, resulting in very large values of disparities. Therefore, the statistics of disparities between the real dataset and the synthetic dataset are different, causing networks to get stuck in local minima when they are fine-tuned on the real data. We handled this issue by incorporating an additional loss function and using the SGM algorithm's valid outputs as pseudo-ground truth during the first few epochs of training. This loss function warms up the training process and resembles a coarse estimation of the ground truth at the beginning of the training. This stratagem prevents the networks from getting stuck in local minima and is used for both CTD and our models.

A qualitative comparison of the estimated disparities of the models on different datasets is depicted in Figure 2.4. It is notable that all of our models produce sharper edges than the baseline model, CTD. Remarkably, our DIS-MF model best preserves the edges and is also capable of retaining high-resolution details. On the other hand, HyperDepth shows poor performance at discontinuities despite its accuracy in smooth regions. The figure also contrasts the quality of our DIS-SF and DIS-FTSF models and exhibits the usefulness of exploiting the DIS-MF model outputs to improve the accuracy of the DIS-SF model. Extended qualitative evaluations are provided in the supplementary material in Section 2.8.6.

Table 2.1 provides the quantitative evaluation of the discussed models and shows the outcomes are consistent with the qualitative results. Table 2.1 also reflects the effect of the dot pattern on the performance of algorithms, where most models have the best accuracy in the experiment with the denser Kinect dot pattern. However, our models show robustness in all experiments. Particularly, DIS-MF yields overall the best results in all the experiments. Also, among the methods that predict disparities based on a single image, our DIS-FTSF model outperforms others overall.

For further experiments and ablation studies of the loss functions, validation masks, components of the DIS-MF network, the effect of imperfect disparities, utilized optical flow network, and extended qualitative analysis, refer to the supplementary material in Section 2.8.

2.7 Conclusion

We proposed DepthInSpace (DIS), which includes three self-supervised deep learning models to estimate depth from structured-light sensor data. Leveraging optical flow, we utilize information from multiple video frames from the same scene to improve depth estimation accuracy in three different self-supervised fashions. We qualitatively and quantitatively evaluated our models over four datasets: a publicly available synthetic dataset, two synthetic datasets customized with our setup parameters and dot pattern, and a real dataset that we made publicly available. The experiments validate the superiority of our models over the existing state-of-the-art methods.

The natural extension for future work will be on the one hand to apply our method to active stereo setup, combining the strengths of both sources of information, and on the other hand to deal with a simplified setup, for instance with a sparser less energy-hungry pattern of illumination.

2.8 Supplementary Material

2.8.1 Notation Review

The notation in this section is consistent with the whole Chapter 2. We name our loss functions as photometric loss $\mathscr{L}_{\boldsymbol{ph}}$, smoothness loss $\mathscr{L}_{\boldsymbol{s}}$, multi-view loss $\mathscr{L}_{\boldsymbol{mv}}$, and pseudo-ground truth loss $\mathscr{L}_{\boldsymbol{pgt}}$. For the DIS-MF model, we assume there are only two frames and we intend to merge Frame j's feature map $\boldsymbol{\phi_j}$ into Frame i's feature map $\boldsymbol{\phi_i}$. $\boldsymbol{X_i}$ represents 3D point cloud of Frame i obtained using imperfect disparities and camera intrinsic parameters. We also define warped features $\boldsymbol{\phi_{j\to i}} = w^{j\to i}(\boldsymbol{\phi_j})$, and warped points $\boldsymbol{X_{j\to i}} = w^{j\to i}(\boldsymbol{X_j})$, where $w^{j\to i}(\cdot)$ stands for bilinear 2D warping via the optical flow $\boldsymbol{F_{i\to j}}$.

2.8.2 Validation Binary Masks for the Multi-View Loss Function

As explained in Section 2.5, our multi-view loss function is defined as:

$$\mathscr{L}_{\boldsymbol{mv}}^{\boldsymbol{ij}} = \left| \left\langle \boldsymbol{X_i} - w^{j\to i}\big(\boldsymbol{T_{j\to i}} \times [\boldsymbol{X_j}, \vec{\mathbf{1}}]\big) \right\rangle_z \right| \times \boldsymbol{M'_{j\to i}} \tag{2.10}$$

where $\boldsymbol{T_{j\to i}} \in \mathbb{R}^{3\times 4}$ is the transformation matrix consisting of ego motion parameters, $\vec{\mathbf{1}}$ is an all one matrix, and $\langle\cdot\rangle_z$ operator returns the depth z of its input 3D vector. In Equation (2.10), $\boldsymbol{M'_{j\to i}}$ is a binary mask map validating warped points and preventing the network from being trained with noisy gradient information. In our DIS-SF model, two criteria must be met in order for a warped point to be indicated as valid. The first one is optical flow forward-backward consistency, suggested by Zou et al. (2018) and Meister et al. (2018):

$$\boldsymbol{M'_{FB}} = |\boldsymbol{F_{i\to j}} + w^{j\to i}(\boldsymbol{F_{j\to i}})|^2 < 0.01 \times (|\boldsymbol{F_{i\to j}}|^2 + |w^{j\to i}(\boldsymbol{F_{j\to i}})|^2) + 0.5 \tag{2.11}$$

which is exactly the binary mask map that is used in the fusion block of our DIS-MF model in Section 2.4.2. The second criterion excludes outlier points whose depths with respect to the camera position of the target frame have a considerable distance from the depths of their corresponding points in the target frame itself, *i.e.*:

$$\boldsymbol{M'_O} = \left| \left\langle \boldsymbol{X_i} - w^{j\to i}\big(\boldsymbol{T_{j\to i}} \times [\boldsymbol{X_j}, \vec{\mathbf{1}}]\big) \right\rangle_z \right| < \tau \tag{2.12}$$

where τ is a threshold set to 10 cm. Accordingly, the binary mask map in our DIS-SF model is defined as the intersection of these two criteria: $\boldsymbol{M'_{j\to i}} = \boldsymbol{M'_{FB}} \odot \boldsymbol{M'_O}$.

Instead, for the DIS-MF model, we define more strict and accurate criteria thanks to having access to imperfect disparities information. After checking optical flow forward-backward consistency in Equation (2.11), we check if the optical flow is consistent with the rigid flow derived from camera motion parameters and imperfect disparities. Therefore, this constraint excludes pixels that either are warped with an erroneous optical flow or contain a significant error in their initially predicted depths:

$$\boldsymbol{M'_{RF}} = \left| \langle \boldsymbol{K} \times \boldsymbol{X_i} \rangle_{uv} - \left\langle w^{j\to i}(\boldsymbol{K} \times \boldsymbol{T_{j\to i}} \times [\boldsymbol{X_j}, \vec{\mathbf{1}}]) \right\rangle_{uv} \right| < 1 \tag{2.13}$$

where $\boldsymbol{K}$ is the camera intrinsic matrix, $\vec{\mathbf{1}}$ is an all-one matrix, and $\langle\cdot\rangle_{uv}$ operator denotes mapping the 3D points on the image plane by dividing the operands by their depth components z. We map the points on the image plane because we are interested in taking into account the validity of the depth of the frame $\boldsymbol{z_j}$ independent of the depth of target frame $\boldsymbol{z_i}$. Thus, if a particular warped point has an accurate initially predicted depth to some degree, but the target point's depth is inaccurate, the warped point is still considered valuable for participating in the loss function, and in fact, it encourages the network to correct the depth of the target point. The threshold in Equation (2.13) is set to 1, and it means that the warped point passes this criterion if it falls into its corresponding matched point's neighborhood of size one pixel on the target image plane.

Lastly, to prevent the network from being greedy in aggregating and fusing and also to encourage it to preserve sharp edges in the disparity map, we introduce another criterion that relates to the visual consistency of the warped and original normalized ambient images. This constraint specially excludes faulty warped points near the edges in the image and encourages the network to preserve the edges and geometry of the objects in the scene:

$$\boldsymbol{M'_{VC}} = |\boldsymbol{A_i} - w^{j\to i}(\boldsymbol{A_j})| < 0.01 \tag{2.14}$$

This criterion is defined on the normalized ambient images, so the 0.01 threshold in Equation (2.14) means points are accepted if their gray intensity levels differ less than one percent of the whole intensity level's range. Having these criteria together, we define the validation binary mask in our DIS-MF model as: $\boldsymbol{M'_{j\to i}} = \boldsymbol{M'_{FB}} \odot \boldsymbol{M'_{RF}} \odot \boldsymbol{M'_{VC}}$. An ablation study on the validation masks of DIS-MF is provided in Experiment 4 of Table 2.6 in Section 2.8.5.

2.8.3 Optical Flow Versus Back-Projection Projection Technique

As we briefly discussed in Section 2.4.1, while it is possible to find matched pixels using the back-projection projection technique, as it is used in CTD (Riegler et al., 2019), we opted for utilizing direct optical flow predictions for finding matched pixels among frames.

Optical flow provides us with two advantages. Firstly, the back-projection projection technique uses the momentary estimated depth and camera motion parameters to find matched pixels. As a result, the errors in the predicted depths propagate through the algorithm and degrade its performance. Also, it causes rapid fluctuations in the optimization landscape of the loss function. In contrast, we use a pre-trained optical flow network that finds matched pixels independently of the performance of depth estimation itself and provides a more reliable and stable loss function to optimize. Secondly, leveraging optical flow enables us to define strict binary masks (described in Section 2.8.2) to exclude incorrectly warped pixels and only retain valuable data for supervising the training.

However, the drawback is that optical flow networks are designed to work with RGB images, while we provide them with ambient images that include only luminance information. To be more specific, we had to convert the ambient images to RGB format before feeding them to LiteFlowNet (Hui et al., 2018). This conversion would degrade optical flow prediction performance, but it does not degrade the performance of our depth estimation models proportionately. The reason is that we have made our models robust to the performance of optical flow thanks to our multiple masking criteria (Section 2.8.2). We only retain correctly warped pixels, so a lower optical flow performance results in fewer pixels for supervision, not more faulty pixels. An ablation study on the effect of the optical flow accuracy on our models' performances is provided in Section 2.8.5.

2.8.4 Implementation Details

Synthetic Datasets: We use the renderer tool provided by Riegler et al. (2019) to generate the synthetic datasets. Following their approach, we populate the scene with a subset of chair meshes from the ShapeNet Core dataset (Chang et al., 2015) randomly scaled and rotated, placed at a distance between 0.5–3 m from the camera. A randomly slanted background plane at a distance between 2–5 m is also employed in the scene. Following the recipe by Riegler et al. (2019), the camera is randomly translated within $20 \times 20 \times 20$ cm for capturing frames of each video sequence. However, we use our own camera's parameters to capture data from the scene. The baseline between our camera and projector is 2.46 cm, and the image resolution is 512×432 pixels. As we already mentioned in Section 2.6, three different dot patterns are exploited in our experiments resulting in three synthetic datasets.

Real Dataset: We use an Artec Eva 3D scanner to scan 4 different 3D scenes containing various objects. The scanner has a few limitations, such as having a limited depth range of 0.3–1.2 m and capturing only 100 depth frames per scan. For this reason, we take multiple partial

Model	λ_1	λ_2	λ_3
DIS-SF	0.4	0.0	0.2
DIS-FTSF	0.4	0.1	0.2
DIS-MF	0.8	0.0	0.2

Table 2.2: The values of the coefficients used in our models' aggregate loss functions in Equation (2.15). For an ablation study of searching for these hyperparameters, refer to Section 2.8.5

scans of each scene and register them together using the point-to-plane variant of the ICP algorithm (Rusinkiewicz and Levoy, 2001), as mentioned in Section 2.6. Once we obtain a 3D point cloud that covers the scene in a suitable way, the Ball-Pivoting algorithm (Bernardini et al., 1999) is applied to derive a 3D mesh of the scene. This mesh is used as ground truth. Subsequently, we scan each scene with our own structured-light sensor to capture 148 pairs of dot images and ambient images. The sensor we use is equipped with a programmable switch, enabling the projector to be on and off, so it can capture dot images and ambient images alternately at the rate of 15 fps each. Given the capturing rate, each pair of a dot image and an ambient image captures the same scene approximately. We make use of these frames as 37 video sequences containing 4 frames each. As a result, 148 video sequences are obtained in total from all 4 scenes. Applying a block-matching technique on the dot images, we extract 148 depth images for each scene. The same ICP algorithm (Rusinkiewicz and Levoy, 2001) is then used to align the depth images acquired using the sensor and the ground truth 3D mesh in order to obtain transformation matrices among the frames. We made the real dataset, along with the instructions to replicate the synthetic datasets, publicly available.

Loss Functions: Our proposed models' loss functions are explained in detail in Section 2.5. Here we clarify the coefficients used in the aggregate loss function. We introduced the aggregate loss function in the paper as:

$$\mathscr{L} = \frac{1}{N}\sum_{i\in\Gamma}(\mathscr{L}^{\boldsymbol{i}}_{\boldsymbol{ph}} + \lambda_1\mathscr{L}^{\boldsymbol{i}}_{\boldsymbol{s}} + \lambda_2\mathscr{L}^{\boldsymbol{i}}_{\boldsymbol{pgt}}) + \frac{1}{N(N-1)}\sum_{i,j\in\Gamma}\lambda_3\mathscr{L}^{\boldsymbol{ij}}_{\boldsymbol{mv}} \tag{2.15}$$

where Γ denotes the image samples representing the same scene, $\mathscr{L}_{\boldsymbol{ph}}$ is the photometric loss, $\mathscr{L}_{\boldsymbol{s}}$ is the smoothness loss, $\mathscr{L}_{\boldsymbol{mv}}$ is the multi-view loss, and $\mathscr{L}_{\boldsymbol{pgt}}$ is the pseudo-ground truth loss. The coefficients in Equation (2.15) take different values for each model. These values are presented in Table 2.2. As is shown in the table, the coefficient of the smoothness loss λ_1 is set differently in our DIS-SF and DIS-MF models. Since our binary masks are less strict in DIS-SF, it allows more pixels in flat regions or backgrounds into $\mathscr{L}_{\boldsymbol{mv}}$. This promotes the smoothness of depth maps, and accordingly, λ_1 is set to a smaller value in DIS-SF. In Section 2.8.5, we provide a detailed ablation study of searching for these hyperparameters.

Training Details: We have trained our models with a single NVIDIA Tesla V100 GPU. We exploit Adam (Kingma and Ba, 2014) optimizer with a learning rate of 10^{-4} for training. We set the

Dataset	CTD (Riegler et al., 2019)	DIS-SF	DIS-FTSF	DIS-MF
Synthetic	40	36	18	85
Real	7	6	3	12

Table 2.3: Approximate time (hours) of training/fine-tuning of the models on synthetic and real datasets with a single NVIDIA Tesla V100 GPU. Note that these reported times are measured assuming all required data are available. Therefore, the dependency of our models on each other's outputs should also be considered. For example, to obtain a trained DIS-FTSF model on a synthetic dataset from scratch, we train the DIS-SF model for 36 hours, the DIS-MF model for 85 hours, and lastly, the DIS-FTSF model for 18 hours approximately.

batch size to 8 for training the DIS-SF and DIS-FTSF models and set it to 4 for the DIS-MF model due to GPU memory limitation.

Prior to training, we pre-save the outputs of LiteFlowNet (Hui et al., 2018) and provide them as the optical flow predictions to our model. Since there are 4 frames in each video sequence of our datasets, we need to pre-save 12 optical flow maps for each sequence. The reason is that we need both forward and backward optical flow data for each unique pair of frames in a video sequence.

We also pre-save the results of our DIS-SF model before training our DIS-MF model, and similarly, we pre-save the outputs of DIS-MF before training the DIS-FTSF model. These pre-saving steps make data preparation more convenient and result in an efficient training process. The time of training of our models, as well as the CTD (Riegler et al., 2019) model, on synthetic datasets and fine-tuning on the real dataset, is presented in Table 2.3.

2.8.5 Ablation Study

This section provides ablation studies regarding our introduced loss functions, setting hyper-parameters, and design choices of our DIS-MF network architecture. All the experiments are evaluated on the synthetic dataset with our projection dot pattern.

Efficacy of the Loss Functions

Here, we attempt to distinguish the effectiveness of each of our individual loss functions. Table 2.4 summarizes the quantitative evaluation of our DIS-SF network when it is trained with different combinations of loss functions. Since we use exactly the same network architecture of CTD (Riegler et al., 2019) in DIS-SF, it is also fair to compare the results with the case that the network is trained with loss functions employed in CTD. Furthermore, to show the generalization of the concept of fine-tuning with pseudo-labels, we fine-tuned the CTD model with our $\mathscr{L}_{pgt}$ as an additional loss function and observed improvement in the outputs as it is indicated in Table 2.4. Lastly, we also examined the performance if we naively supervised the

Loss	$o(0.5)$	$o(1)$	$o(2)$	$o(5)$
Supervised by SGM Outputs	21.1	15.9	12.0	7.91
CTD Loss Functions	3.38	1.71	0.85	0.28
CTD Loss Functions + $\mathscr{L}_{pgt}$	2.62	1.25	0.58	0.14
$\mathscr{L}_{ph}$	40.5	22.1	9.36	1.15
$\mathscr{L}_{ph} + \mathscr{L}_{s}$	3.52	1.38	0.67	0.23
$\mathscr{L}_{ph} + \mathscr{L}_{mv}$	3.22	1.69	0.86	0.24
$\mathscr{L}_{ph} + \mathscr{L}_{s} + \mathscr{L}_{mv}$	2.31	1.24	0.62	0.19
$\mathscr{L}_{ph} + \mathscr{L}_{s} + \mathscr{L}_{mv} + \mathscr{L}_{pgt}$	**1.96**	**0.95**	**0.45**	**0.12**

Table 2.4: Ablation study of individual loss terms: the photometric loss $\mathscr{L}_{ph}$, the smoothness loss $\mathscr{L}_{s}$, the multi-view loss $\mathscr{L}_{mv}$, and the pseudo-ground truth loss $\mathscr{L}_{pgt}$. Since we share the same network architecture for single-frame depth estimation with CTD (Riegler et al., 2019), we could fairly contrast the superiority of our loss functions to that of CTD. The last row, which represents our DIS-FTSF model, yields overall the best results in terms of the percentage of outliers. Also, the effectiveness and generalizability of fine-tuning with pseudo-ground truth labels $\mathscr{L}_{pgt}$ are shown when used for either fine-tuning our DIS-SF model or the CTD model.

network with valid outputs resulting from the SGM algorithm (Hirschmuller, 2007).

Figure 2.5 also depicts some examples of our DIS-SF network outputs when it is trained with different combinations of loss functions. The samples demonstrate how our loss functions complement each other and form a robust depth estimation model altogether.

Searching for Hyperparameters

Table 2.5 represents the experiments we conducted for searching for the hyperparameters of our aggregate loss function for the DIS-SF and DIS-MF models. As we previously discussed, the hyperparameters should take different values due to the different mask criteria we defined in Section 2.8.2. In the DIS-SF model, the mask criteria are less strict, and as a result, the smoothness of disparity maps is inherently promoted in the multi-view loss $\mathscr{L}_{mv}$, whereas smoothness loss $\mathscr{L}_{s}$ in DIS-MF is emphasized directly. The experiments in Table 2.5 support this hypothesis.

DIS-MF Network Architecture

The quantitative analysis of four experiments is presented in Table 2.6: 1. Effect of the number of fused frames. 2. Effect of the number of fusion blocks and their channel size on the overall network architecture. 3. Efficacy of various processing elements in our DIS-MF network. 4. Contribution of validation binary masks, introduced in Section 2.8.2, on our model's performance. Moreover, qualitative analysis of these four experiments are provided in Figures 2.6, 2.7, 2.8, and 2.9.

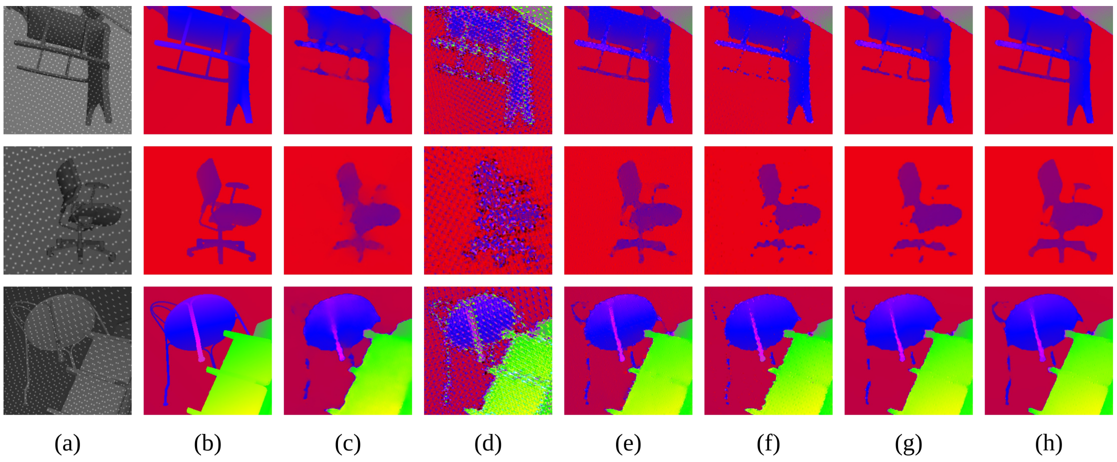


Figure 2.5: Qualitative ablation study of our individual loss functions. For each experiment, our DIS-SF network is trained with a different combination of our loss functions. (a) Input dot image with the projected pattern. (b) Ground truth disparity map. (c) Trained with CTD (Riegler et al., 2019) loss functions. (d) Trained with $\mathscr{L}_{ph}$. (e) Trained with $\mathscr{L}_{ph} + \mathscr{L}_{s}$. (f) Trained with $\mathscr{L}_{ph} + \mathscr{L}_{mv}$. (g) Trained with $\mathscr{L}_{ph} + \mathscr{L}_{s} + \mathscr{L}_{mv}$. (h) Our final single-frame model trained with $\mathscr{L}_{ph} + \mathscr{L}_{s} + \mathscr{L}_{mv} + \mathscr{L}_{pgt}$.

In particular, it is notable that despite the hugeness of our network and its robustness against variation of the number of fusion blocks or their channel size, according to Table 2.6, the performance still degrades when we remove continuous 3D convolution and perform fusion only on the 2D grid map. As previously discussed, in the context of depth estimation from conventional RGB camera, works like DeepV2D (Teed and Deng, 2019), DeepMVS (Huang et al., 2018), DeepSFM (Wei et al., 2020), and DPSNet (Im et al., 2018) attempt to fuse information of multiple frames on the 2D grid. We cannot directly compare our work with them because they are designed to operate on RGB images. Still, we attempt to demonstrate how leveraging fusion both on the 2D grid and in the 3D space improves the performance in the structured-light setup depth estimation.

Lastly, Figure 2.10 exemplifies the behavior of each binary mask criterion in Section 2.8.2 and shows how these masks exclude low confidence pixels and prevent them from supervising the training.

Utilizing the Fusion Architecture in Different Setups

This section analyzes the effect of three factors on our DIS-MF model's performance: the imperfect input disparities, the fusion architecture, and our proposed multi-view-based loss function. In this regard, we applied our DIS-MF network to CTD outputs and trained the fusion model once with the CTD loss function and once with ours. Comparison of the results with the original CTD and DIS models in Table 2.7 discriminates the individual efficacy of our

Model	$\lambda_1(\mathscr{L}_s)$	$\lambda_3(\mathscr{L}_{mv})$	$o(0.5)$	$o(1)$	$o(2)$	$o(5)$
DIS-SF	0.2	0.1	2.84	1.45	0.68	0.20
	0.2	0.2	2.48	1.31	0.64	0.19
	0.2	0.3	2.41	1.35	0.70	0.19
	0.2	0.4	2.35	1.26	0.66	0.19
	0.4	0.1	2.34	1.26	0.63	0.19
	0.4	0.2	**2.31**	**1.24**	**0.62**	**0.19**
	0.4	0.3	2.37	1.28	0.64	0.19
	0.4	0.4	2.49	1.29	0.67	0.20
	0.6	0.1	2.37	1.27	0.65	0.19
	0.6	0.2	2.32	1.24	0.63	0.19
	0.6	0.3	2.42	1.28	0.65	0.19
	0.6	0.4	2.44	1.31	0.68	0.19
	0.8	0.1	2.35	1.25	0.65	0.19
	0.8	0.2	2.40	1.28	0.66	0.19
	0.8	0.3	2.43	1.30	0.67	0.20
	0.8	0.4	2.50	1.38	0.69	0.20
DIS-MF	0.2	0.1	2.19	0.93	0.43	0.12
	0.2	0.2	1.98	0.93	0.44	0.12
	0.2	0.3	1.99	0.96	0.47	0.13
	0.4	0.1	1.92	0.86	0.39	0.11
	0.4	0.2	1.87	0.84	0.39	0.11
	0.4	0.3	1.81	0.88	0.42	0.11
	0.6	0.1	2.06	0.86	0.36	0.10
	0.6	0.2	1.97	0.84	0.37	0.10
	0.6	0.3	1.70	0.80	0.38	0.11
	0.8	0.1	1.96	0.82	0.35	0.10
	0.8	0.2	**1.58**	**0.71**	**0.32**	**0.10**
	0.8	0.3	1.66	0.77	0.36	0.10
	1.0	0.1	1.88	0.75	0.34	0.10
	1.0	0.2	1.72	0.75	0.34	0.10
	1.0	0.3	1.62	0.76	0.35	0.10

Table 2.5: Ablation study of searching for hyperparameters of our aggregate loss function. It is noticeable that even poor choices of hyperparameters for the DIS-MF model still result in better performance than all experiments with the DIS-SF model. For further details on why separate experiments were needed for the DIS-SF and DIS-MF models, refer to Section 2.8.5 and 2.8.4.

proposed loss and fusion architecture and shows these two contributions are complementary to each other. Also, the table suggests that these two are not necessarily required to be applied to DIS-SF imperfect disparities, as they can significantly improve the quality of CTD outputs.

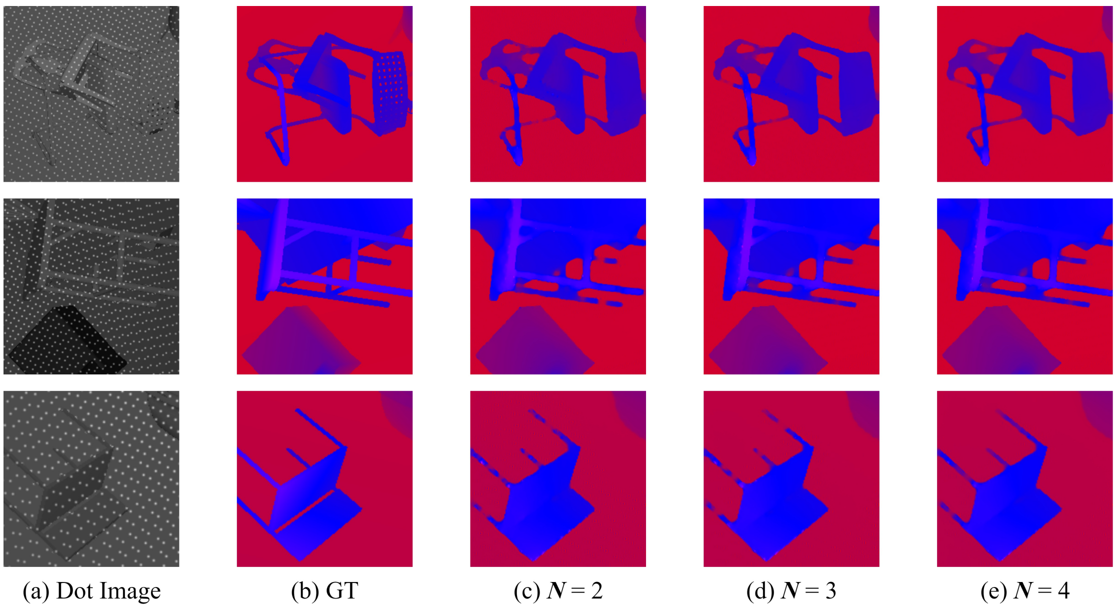


Figure 2.6: Qualitative analysis of Experiment 1 in Table 2.6, where we examine the effect of the number of fused frame N on DIS-MF performance. (a) Input dot image. (b) Ground truth disparity map. (c) DIS-MF's output with $N = 2$. (d) DIS-MF's output with $N = 3$. (e) DIS-MF's output with $N = 4$. As expected according to Table 2.6, DIS-MF performs better at completing disparity maps as the number of fused frames increases.

Robustness to Optical Flow Predictions

As discussed in Section 2.8.3, our training process is robust to optical flow prediction errors thanks to our validation masks introduced in Section 2.8.2. This robustness allows us to utilize the lightweight optical flow network LiteFlowNet (Hui et al., 2018) in our models. To evaluate this robustness, we also trained our networks on the synthetic data with the state-of-the-art optical flow model on MPI Sintel (Butler et al., 2012), GMA (Jiang et al., 2021), which has **69.4**% higher accuracy but is slower than LiteFlowNet. The results in Table 2.8 show that despite the huge gap between the GMA and LiteFlowNet accuracies, the gain that GMA brings to our models is marginal. It is also notable that since our validation masks in DIS-MF are stricter than in DIS-SF on account of having access to imperfect disparities (see Section 2.8.2), DIS-MF is more robust to the optical flow performance.

2.8.6 Additional Qualitative Results

In this section, we first present an extended qualitative analysis of existing models along with our proposed models in Figures 2.11, 2.12, 2.13, and 2.14. Each figure presents sample images from one of the datasets introduced in Section 2.6. The color bars in the figures represent disparity error maps in pixels.

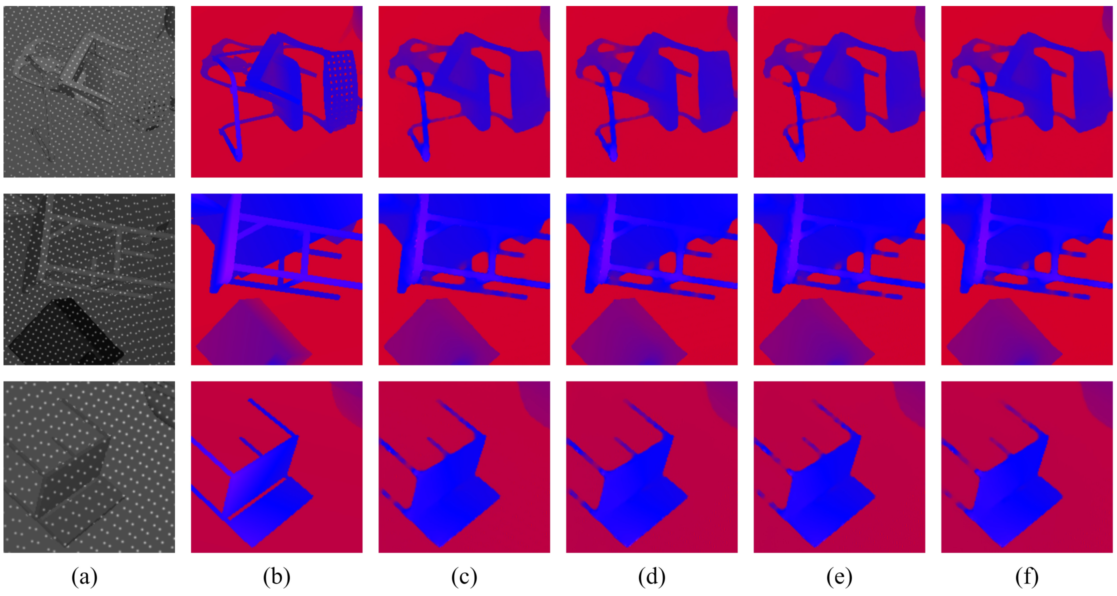


Figure 2.7: Qualitative analysis of Experiment 2 in Table 2.6, where we examine the effect of the number of fusion blocks and their channel size on DIS-MF performance (The architecture is presented in Figure 2.3). (a) Input dot image. (b) Ground truth disparity map. (c) DIS-MF's output with 2 cascaded fusion blocks, each of which with 48 channels. (d) DIS-MF's output with 4 cascaded fusion blocks, each of which with 32 channels. (e) DIS-MF's output with 6 cascaded fusion blocks, each of which with 24 channels. (f) DIS-MF's output with 8 cascaded fusion blocks, each of which with 16 channels. As expected according to Table 2.6, DIS-MF performs robustly with different choices of parameters, and having 4 fusion blocks with 32 channels produces higher quality disparity maps.

Furthermore, we adopt a different approach to visualize depth maps of objects from the synthetic dataset in Figure 2.15 and from the real dataset in Figure 2.16. To visualize the depth maps, we rely on color-coded depth values and 3D rendering of the scene. The depths are rendered in OpenGL (Shreiner et al., 2013), where every point of the mesh has a color that corresponds to its depth value in the Turbo colormap (Mikhailov, 2019). In addition, we place a spotlight in the same position as the camera to light the scene. This allows us to better appreciate the quality of the computed depths as small changes in the values of the normals in the mesh impact the lighting. Finally, we consider that edges over 10 cm in the mesh are invalid and exclude them from the rendered images.

$\boldsymbol{N}$	Fusion Blocks		Network Components			Binary Masks Criteria			Metrics [%]				Examples
	Num.	Ch.	2D Fus.	3D Fus.	Ref.	$\boldsymbol{M'_{FB}}$	$\boldsymbol{M'_{RF}}$	$\boldsymbol{M'_{VC}}$	$o(0.5)$	$o(1)$	$o(2)$	$o(5)$	
Experiment 1: Effect of the number of fused frames $\boldsymbol{N}$.													
2	4	32	●	●	●	●	●	●	1.90	0.83	0.37	0.10	Figure 2.6.c
3	4	32	●	●	●	●	●	●	1.73	0.75	0.34	0.10	Figure 2.6.d
4	4	32	●	●	●	●	●	●	**1.58**	**0.71**	**0.32**	**0.10**	Figure 2.6.e
Experiment 2: Effect of the number of fusion blocks and their channel size.													
4	2	48	●	●	●	●	●	●	1.59	0.72	0.33	0.10	Figure 2.7.c
4	4	32	●	●	●	●	●	●	**1.58**	**0.71**	**0.32**	**0.10**	Figure 2.7.d
4	6	24	●	●	●	●	●	●	1.62	0.72	0.33	0.10	Figure 2.7.e
4	8	16	●	●	●	●	●	●	1.60	0.73	0.33	0.10	Figure 2.7.f
Experiment 3: Each processing component's efficacy in the network architecture.													
4	4	32	○	●	●	●	●	●	1.58	0.72	0.33	0.10	Figure 2.8.c
4	4	32	●	○	●	●	●	●	1.74	0.74	0.34	0.11	Figure 2.8.d
4	4	32	●	●	○	●	●	●	1.98	1.04	0.50	0.13	Figure 2.8.e
4	4	32	●	●	●	●	●	●	**1.58**	**0.71**	**0.32**	**0.10**	Figure 2.8.f
Experiment 4: Impact of the validation binary masks on the performance.													
4	4	32	●	●	●	●	○	○	2.19	1.11	0.53	0.14	Figure 2.9.c
4	4	32	●	●	●	●	○	●	1.63	0.71	0.33	0.10	Figure 2.9.d
4	4	32	●	●	●	●	●	○	1.88	0.91	0.42	0.11	Figure 2.9.e
4	4	32	●	●	●	●	●	●	**1.58**	**0.71**	**0.32**	**0.10**	Figure 2.9.f

Table 2.6: Ablation study of DIS-MF network. This table summarizes four experiments, in each of which the effect of a subset of particular design choices is examined independently. Firstly, this study shows how the performance of DIS-MF improves with higher numbers of fused frames $\boldsymbol{N}$. The second experiment examines how many cascaded fusion blocks and with which channel size leads to better results. The channel size and the number of blocks are selected such that computation resources become comparable. As the results suggest, our proposed architecture is robust against variations of these parameters. The third experiment evaluates the efficacy of processing components in our network architecture. Our fusion block contains aggregating convolution modules both on the 2D grid map and in the continuous 3D space. Here, the contribution of each part to the final performance is shown (*e.g.*, the $o(0.5)$ metric deteriorates from **1.58** % to **1.74** % when we perform aggregation only on the 2D grid map). For further details on why this comparison is important, please refer to Section 2.8.5. Also, the effectiveness of the refinement structure in the post-processing part of our network architecture is investigated in this experiment. Lastly, this study demonstrates how each of the binary mask criteria in Section 2.8.2 affects our DIS-MF model's performance in the fourth experiment. Qualitative analyses corresponding to each of these experiments are provided in Figures 2.6, 2.7, 2.8, and 2.9.

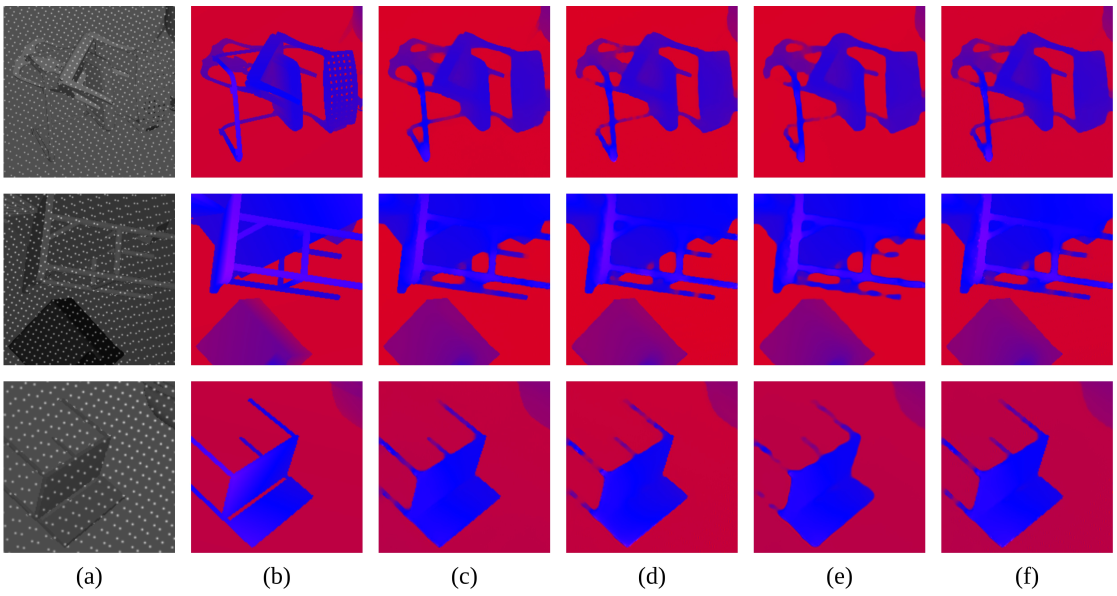


Figure 2.8: Qualitative analysis of Experiment 3 in Table 2.6, where we examine the efficacy of each processing component on DIS-MF performance (Components are introduced in Section 2.4.2). (a) Input dot image. (b) Ground truth disparity map. (c) DIS-MF's output when fusion is done only in the continuous 3D space. (d) DIS-MF's output when fusion is done only on the 2D grid map. (e) DIS-MF's output when the refinement structure (post-processing part in Figure 2.3) is removed from the network architecture. (f) DIS-MF's output with all processing components. This figure, in particular, contrasts the capability of DIS-MF in completing disparity maps when fusion is performed in the continuous 3D space versus when it is done on the 2D grid map. Additionally, by comparing columns (c) and (f), one observes that fusing only in the 3D space results in fewer missing points compared to having both 2D and 3D fusion in the architecture. However, the edge-fattening artifact, which is inherent to continuous 3D convolution, is more pronounced in the disparity maps generated by aggregating data only in the 3D space. Overall, employing both 2D and 3D fusion components produces balanced disparity maps concerning completing missing points and preserving edges and discontinuities in the disparity maps.

Model	$o(0.5)$	$o(1)$	$o(2)$	$o(5)$
CTD	3.38	1.71	0.85	0.28
DIS-SF	2.31	1.24	0.62	0.19
CTD Output + DIS-MF Network + CTD Loss	2.33	1.14	0.60	0.20
CTD Output + DIS-MF Network + DIS Loss	1.97	0.83	0.37	0.12
DIS-MF	**1.58**	**0.71**	**0.32**	**0.10**

Table 2.7: Analysis of our multi-frame fusion efficacy when it takes CTD outputs as imperfect input disparities.

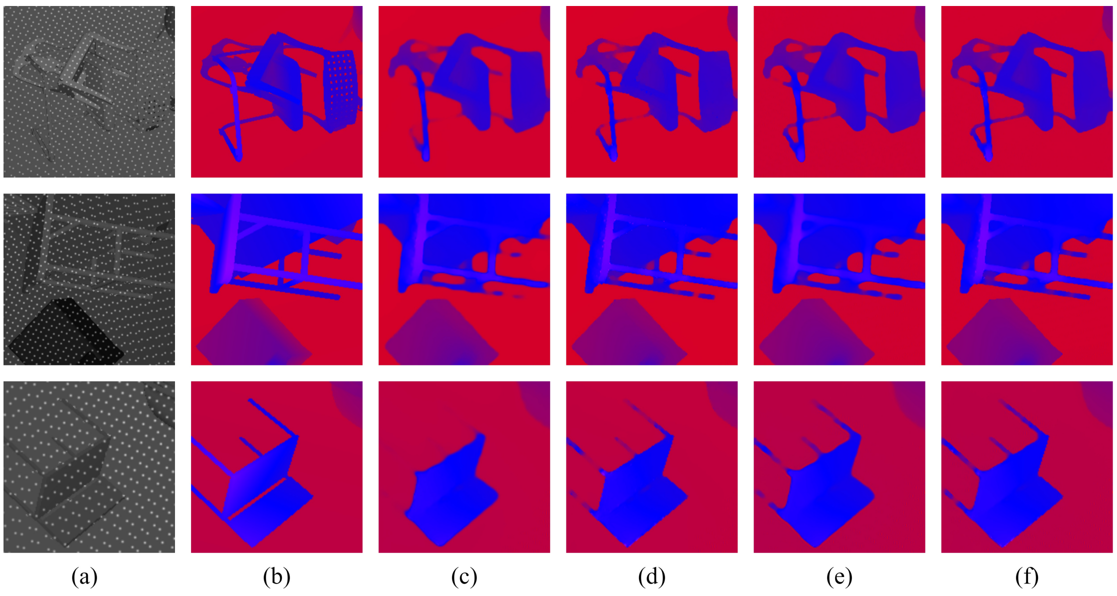


Figure 2.9: Qualitative analysis of Experiment 4 in Table 2.6, where we examine the impact of validation binary masks, introduced in Section 2.8.2, on DIS-MF performance (a) Input dot image. (b) Ground truth disparity map. (c) DIS-MF's output when only $\boldsymbol{M}'_{FB}$ is used to select pixels for $\mathscr{L}_{mv}$. (d) DIS-MF's output when $\boldsymbol{M}'_{FB}$ and $\boldsymbol{M}'_{VC}$ are used to select pixels for $\mathscr{L}_{mv}$. (e) DIS-MF's output when $\boldsymbol{M}'_{FB}$ and $\boldsymbol{M}'_{RF}$ are used to select pixels for $\mathscr{L}_{mv}$. (f) DIS-MF's output when all $\boldsymbol{M}'_{FB}$, $\boldsymbol{M}'_{RF}$, and $\boldsymbol{M}'_{VC}$ are used to select pixels for $\mathscr{L}_{mv}$. It is noticeable how $\boldsymbol{M}'_{RF}$ contributes to completing the disparity map while $\boldsymbol{M}'_{VC}$ properly preserves the edges and discontinuities.

Model	$o(0.5)$	$o(1)$	$o(2)$	$o(5)$
DIS-SF + LiteFlowNet	2.31	1.24	0.62	0.19
DIS-SF + GMA	**2.14**	**1.12**	**0.56**	**0.14**
DIS-MF + LiteFlowNet	**1.58**	**0.71**	0.32	0.10
DIS-MF + GMA	1.62	**0.71**	**0.31**	**0.09**

Table 2.8: Analysis of the effect of using different optical flow networks, LiteFlowNet (Hui et al., 2018) and GMA (Jiang et al., 2021), on DIS models' performances.

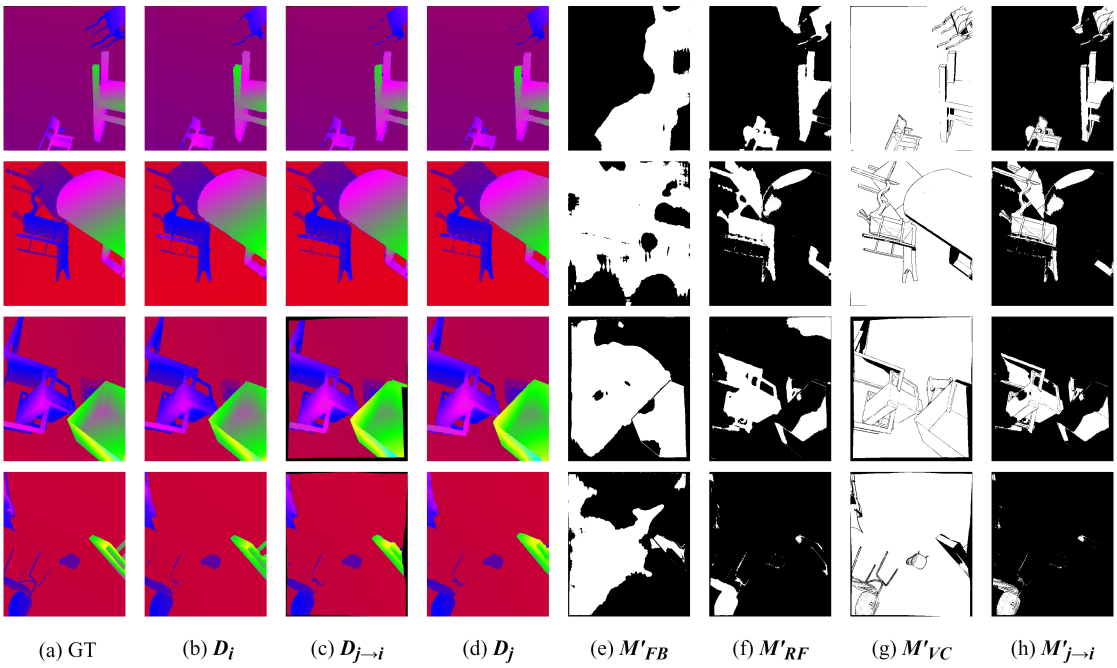


Figure 2.10: Visualizing the behavior of binary masks criteria in Section 2.8.2 when Frame j is being used for supervision and is warped on the Frame i's grid map. This figure demonstrates how each criterion selects high-confidence pixels for training and prevents low-confidence pixels from participating in the supervision and destabilizing training. (a) Frame i's ground truth disparity map. (b) Frame i's imperfect disparity map ($\boldsymbol{D_i}$). (c) Frame j's imperfect disparity map warped on the Frame i's grid map ($\boldsymbol{D_{j\to i}}$). (d) Frame j's imperfect disparity map ($\boldsymbol{D_j}$). (e) Optical flow forward-backward consistency mask $\boldsymbol{M'_{FB}}$. (f) Optical flow and rigid flow consistency mask $\boldsymbol{M'_{RF}}$. (g) Visual consistency of ambient images mask $\boldsymbol{M'_{VC}}$. (h) The intersection of all binary masks $\boldsymbol{M'_{j\to i}} = \boldsymbol{M'_{FB}} \odot \boldsymbol{M'_{RF}} \odot \boldsymbol{M'_{VC}}$.

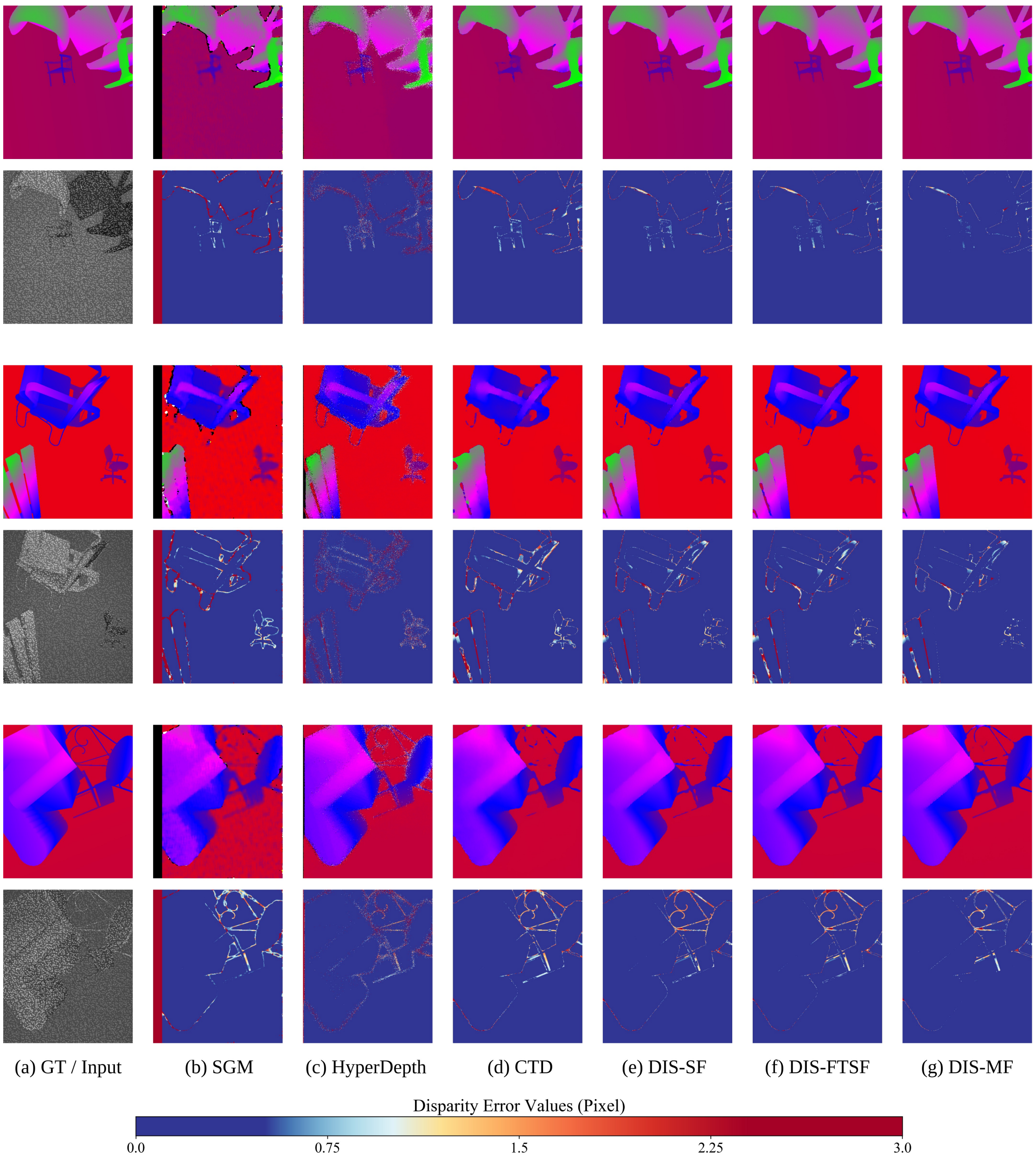


Figure 2.11: Additional full-size qualitative results of the implemented methods and their corresponding error maps. All samples are taken from the synthetic dataset rendered with the Kinect dot pattern. (a) Ground truth disparity map and input dot image. (b) The SGM algorithm (Hirschmuller, 2007). (c) HyperDepth (Ryan Fanello et al., 2016). (d) CTD (Riegler et al., 2019). (e) Our DepthInSpace Single-Frame (DIS-SF) model. (f) Our DepthInSpace Fine-Tuned Single-Frame (DIS-FTSF) model. (g) Our DepthInSpace Multi-Frame (DIS-MF) Model. The color bar represents the disparity error map in pixels.

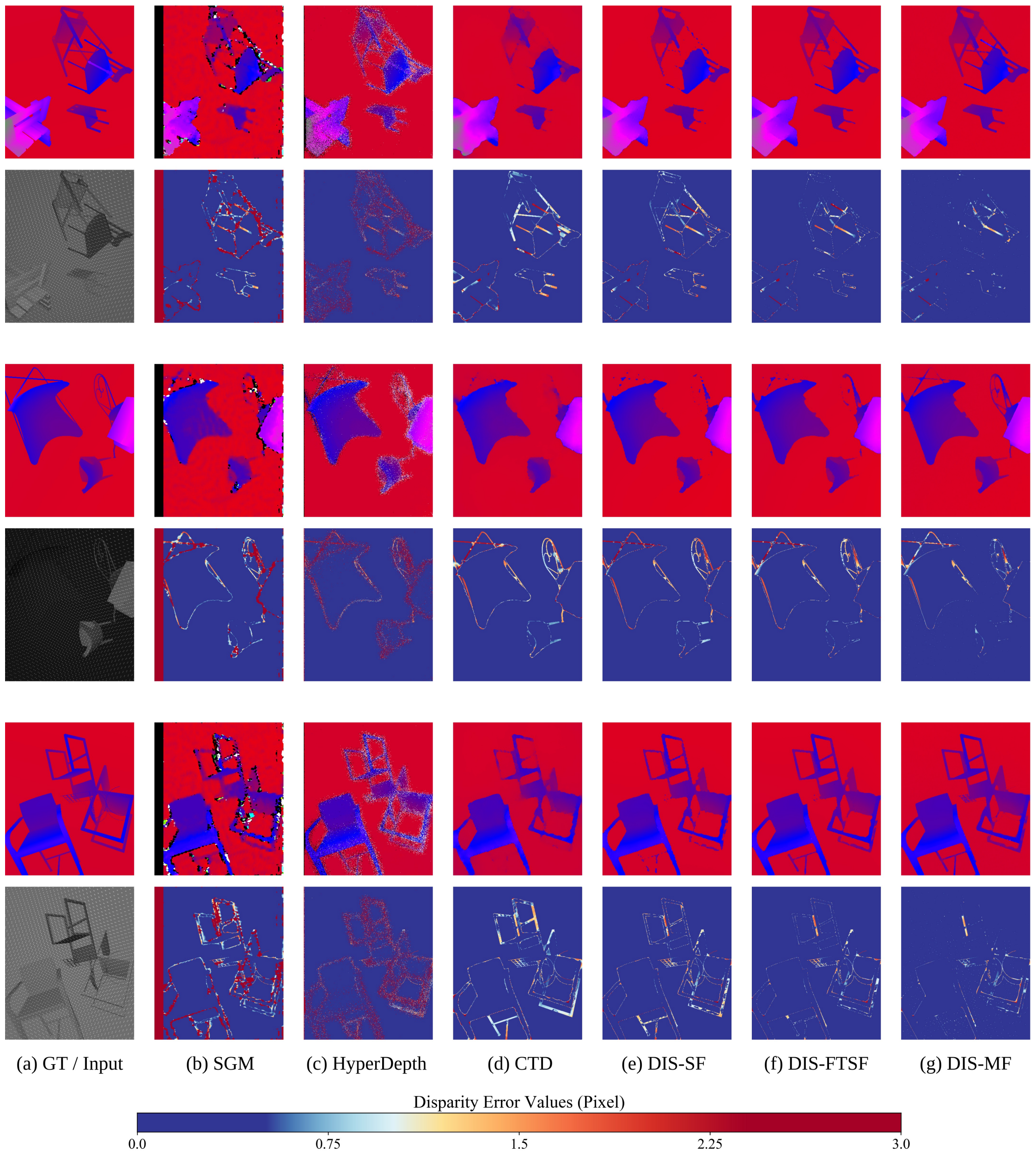


Figure 2.12: Additional full-size qualitative results of the implemented methods and their corresponding error maps. All samples are taken from the synthetic dataset rendered with our own theoretical dot pattern. (a) Ground truth disparity map and input dot image with the projected pattern. (b) The SGM algorithm (Hirschmuller, 2007). (c) HyperDepth (Ryan Fanello et al., 2016). (d) CTD (Riegler et al., 2019). (e) Our DepthInSpace Single-Frame (DIS-SF) model. (f) Our DepthInSpace Fine-Tuned Single-Frame (DIS-FTSF) model. (g) Our DepthInSpace Multi-Frame (DIS-MF) Model. The color bar represents the disparity error map in pixels.

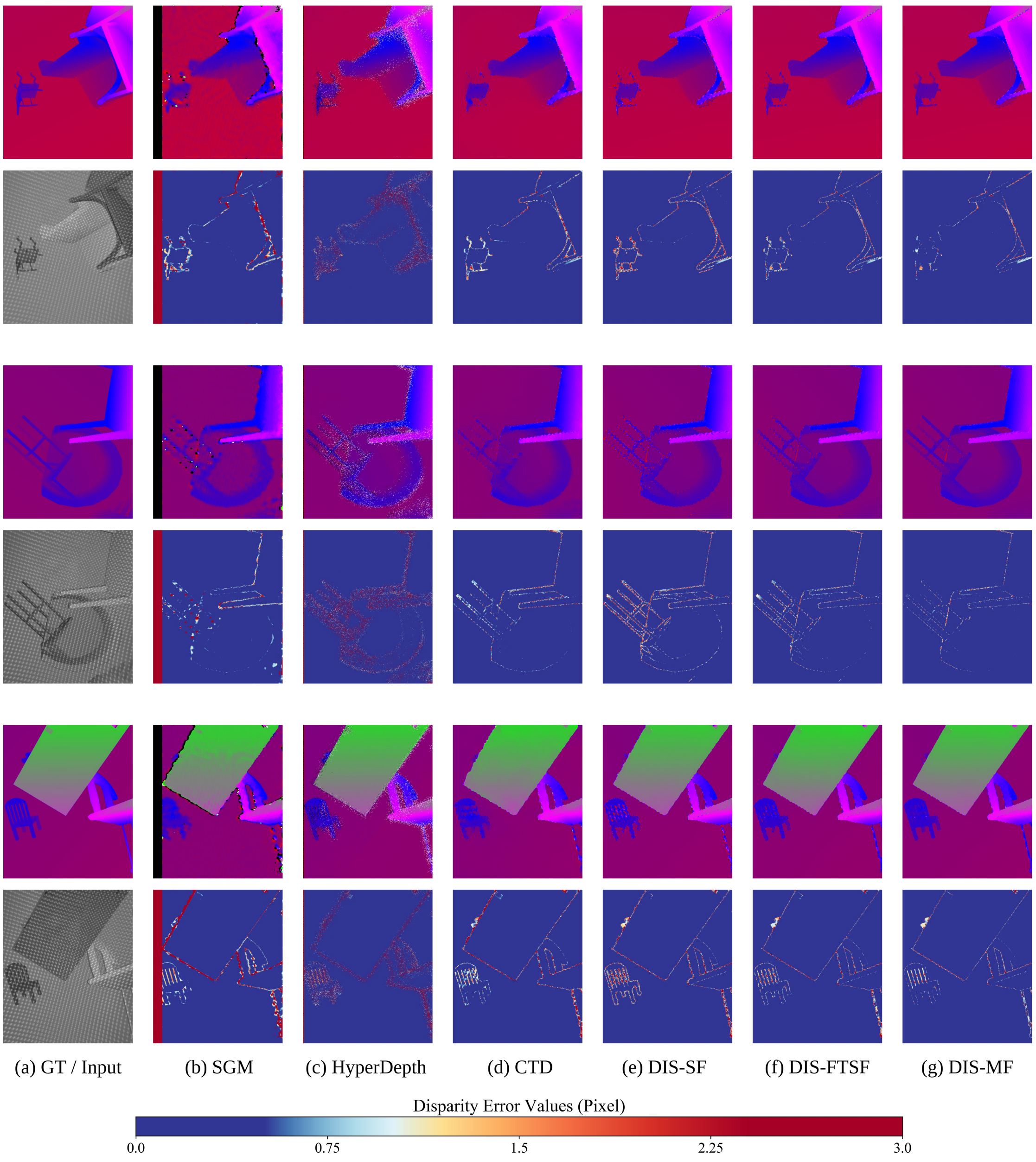


Figure 2.13: Additional full-size qualitative results of the implemented methods and their corresponding error maps. All samples are taken from the synthetic dataset rendered with our own dot pattern observed in an actual laboratory setup. (a) Ground truth disparity map and input dot image. (b) The SGM algorithm (Hirschmuller, 2007). (c) HyperDepth (Ryan Fanello et al., 2016). (d) CTD (Riegler et al., 2019). (e) Our DepthInSpace Single-Frame (DIS-SF) model. (f) Our DepthInSpace Fine-Tuned Single-Frame (DIS-FTSF) model. (g) Our DepthInSpace Multi-Frame (DIS-MF) Model. The color bar represents the disparity error map in pixels.

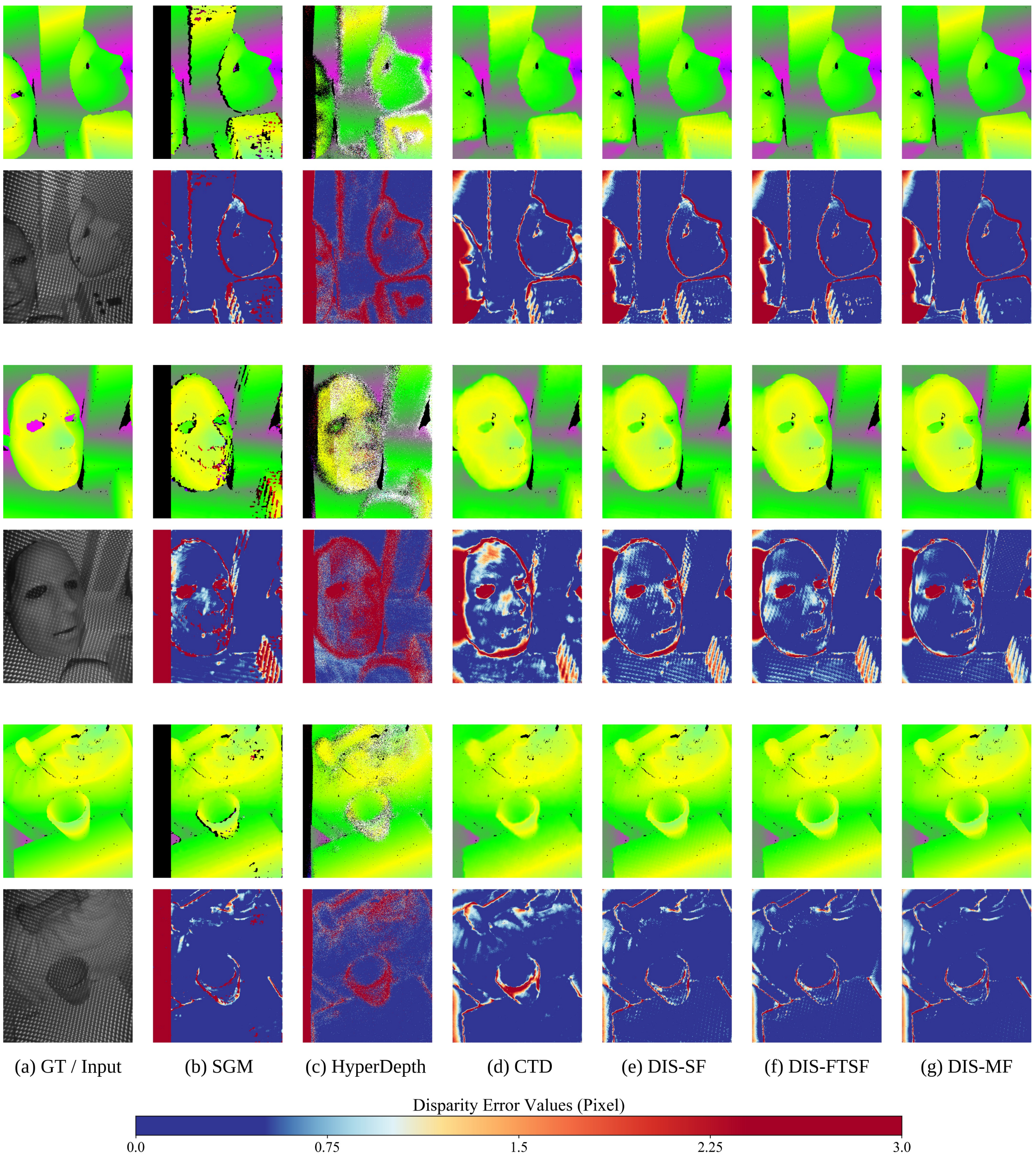


Figure 2.14: Additional full-size qualitative results of the implemented methods and their corresponding error maps. All samples are taken from the real dataset. (a) Ground truth disparity map and input dot image. (b) The SGM algorithm (Hirschmuller, 2007). (c) HyperDepth (Ryan Fanello et al., 2016). (d) CTD (Riegler et al., 2019). (e) Our DepthInSpace Single-Frame (DIS-SF) model. (f) Our DepthInSpace Fine-Tuned Single-Frame (DIS-FTSF) model. (g) Our DepthInSpace Multi-Frame (DIS-MF) Model. Points for which the ground truth data is unavailable are excluded from evaluation. The color bar represents the disparity error map in pixels.

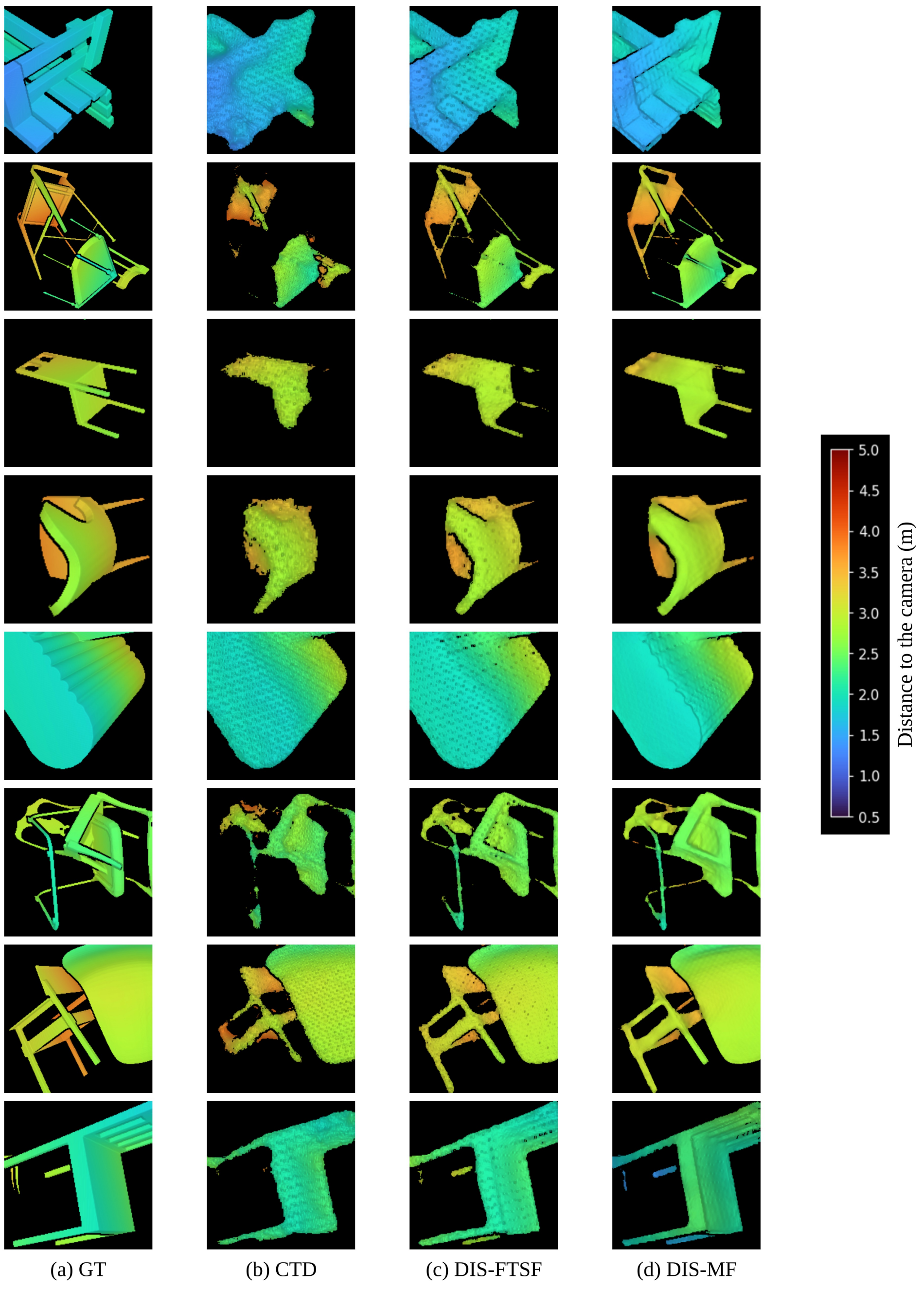


Figure 2.15: Qualitative analysis of the depth maps rendered in OpenGL (Shreiner et al., 2013) and presented in the Turbo color map (Mikhailov, 2019). Samples are taken from the synthetic dataset. This analysis shows how our models outperform the state-of-the-art model, CTD (Riegler et al., 2019), in preserving details of the 3D objects and producing sharp edges. (a) Ground truth depth map. (b) CTD (Riegler et al., 2019). (c) Our DIS-FTSF model. (d) Our DIS-MF model. For further information about 3D rendering of depth maps, refer to Section 2.8.6. The color bar represents depth values in meters.

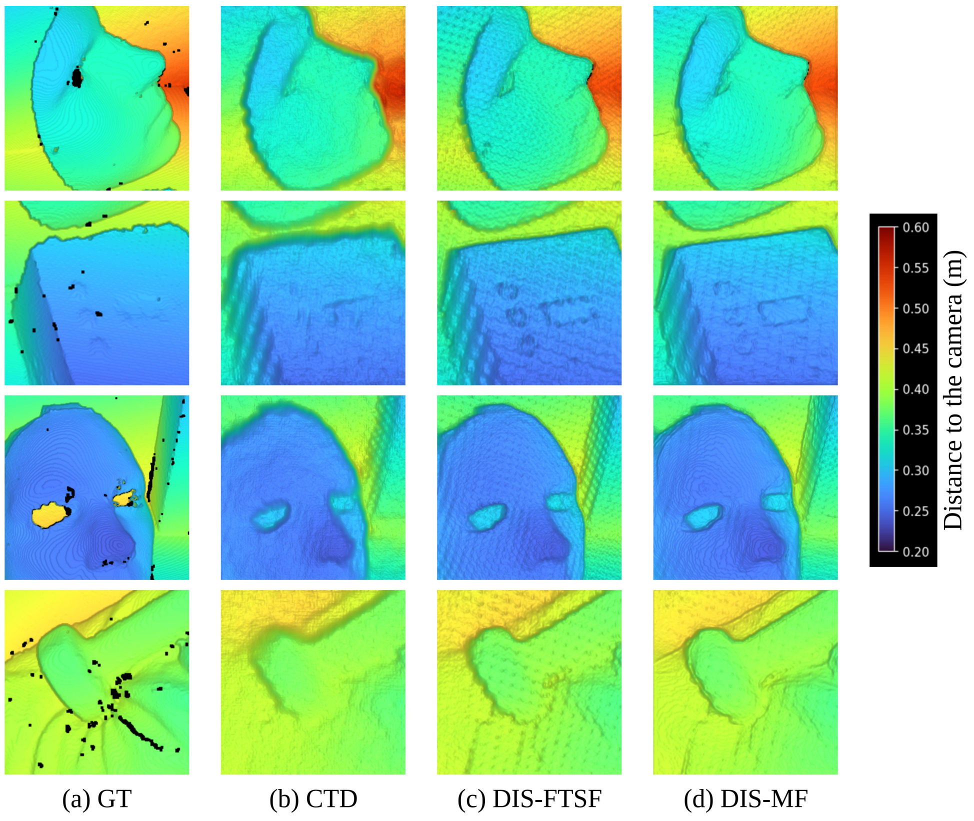


Figure 2.16: Qualitative analysis of the depth maps rendered in OpenGL (Shreiner et al., 2013) and presented in the Turbo color map (Mikhailov, 2019). Samples are taken from the real dataset. This analysis shows how our models outperform the state-of-the-art model, CTD (Riegler et al., 2019), in preserving details of the 3D objects and producing sharp edges. It is noticeable that in some regions (*e.g.*, the top edge of the box in the second row in column (a)), ground truth depths are noisy, and it is due to the limitations of the 3D scanner we used to capture ground truth depths. Since all evaluated methods are self-supervised, their performances are not affected by the ground truth noise. (a) Ground truth depth map. (b) CTD (Riegler et al., 2019). (c) Our DIS-FTSF model. (d) Our DIS-MF model. For further information about 3D rendering of depth maps, refer to Section 2.8.6. The color bar represents depth values in meters.

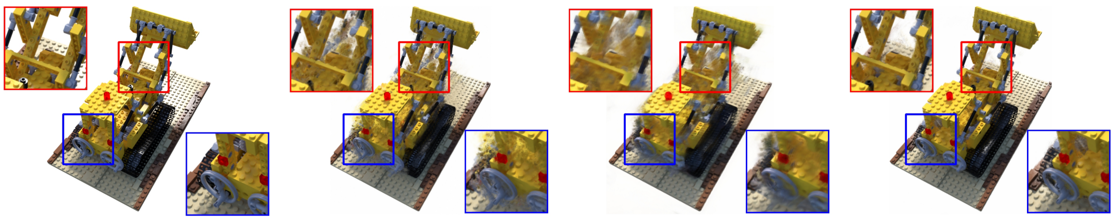
3 GeoNeRF: Generalizing NeRF with Geometry Priors

Disclaimer: This chapter is adapted from the following article – with permission of all co-authors and the conference:

Johari, M. M., Lepoittevin, Y., and Fleuret, F. (2022). GeoNeRF: Generalizing NeRF with Geometry Priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18365–18375.

3.1 Abstract

We present GeoNeRF, a generalizable photorealistic novel view synthesis method based on neural radiance fields. Our approach consists of two main stages: a geometry reasoner and a renderer. To render a novel view, the geometry reasoner first constructs cascaded cost volumes for each nearby source view. Then, using a Transformer-based attention mechanism and the cascaded cost volumes, the renderer infers geometry and appearance and renders detailed images via classical volume rendering techniques. This architecture, in particular, allows sophisticated occlusion reasoning, gathering information from consistent source views. Moreover, our method can easily be fine-tuned on a single scene and renders competitive results with per-scene optimized neural rendering methods with a fraction of computational cost. Experiments show that GeoNeRF outperforms state-of-the-art generalizable neural rendering models on various synthetic and real datasets. Lastly, with a slight modification to the geometry reasoner, we also propose an alternative model that adapts to RGBD images. This model directly exploits the depth information often available thanks to depth sensors. The implementation source code and visualization videos showcasing the obtained results can be accessed through the following link: https://www.idiap.ch/paper/geonerf.



(a) Ground Truth (b) IBRNet (c) MVSNeRF (d) GeoNeRF (ours)

Figure 3.1: Our generalizable GeoNeRF model infers complex geometries of objects in a novel scene without per-scene optimization and synthesizes novel images of higher quality than the existing works: IBRNet (Wang et al., 2021b) and MVSNeRF (Chen et al., 2021).

3.2 Introduction

Novel view synthesis is a long-standing task in computer vision and computer graphics. Neural Radiance Fields (NeRF) (Mildenhall et al., 2020) made a significant impact on this research area by implicitly representing the 3D structure of the scene and rendering high-quality novel images. Our work addresses the main drawback of NeRF, which is the requirement to train from scratch for every scene separately. The per-scene optimization of NeRF is lengthy and requires densely captured images from each scene.

Approaches like pixelNeRF (Yu et al., 2021), GRF (Trevithick and Yang, 2021), MINE (Li et al., 2021a), SRF (Chibane et al., 2021), IBRNet (Wang et al., 2021b), MVSNeRF (Chen et al., 2021), and NeRFormer (Reizenstein et al., 2021) address this issue and generalize NeRF rendering technique to unseen scenes. The common motivation behind such methods is to condition the NeRF renderer with features extracted from source images from a set of nearby views. Despite the generalizability of these models to new scenes, their understanding of the scene geometry and occlusions is limited, resulting in undesired artifacts in the rendered outputs. MVSNeRF (Chen et al., 2021) constructs a low-resolution 3D cost volume inspired by MVS-Net (Yao et al., 2018), which is widely used in the Multi-View Stereo research, to condition and generalize the NeRF renderer. However, it has difficulty rendering detailed images and does not deal with occlusions in a scene. In this study, we take MVSNeRF as a baseline and propose the following improvements.

- We introduce a geometry reasoner in the form of cascaded cost volumes (Section 3.4.1) and train it in a semi-supervised fashion (Section 3.4.4) to obtain fine and high-resolution priors for conditioning the renderer.
- We combine an attention-based model that deals with information coming from different source views at any point in space, by essence permutation invariant, with an auto-encoder network which aggregates information along a ray, leveraging its strong Euclidean and ordering structure (Section 3.4.3).

- Thanks to the symmetry and generalizability of our geometry reasoner and renderer, we detect and exclude occluded views for each point in space and use the remaining views for processing that point (Section 3.4.3).

In addition, with a slight modification to the architecture, we propose an alternate model that takes RGBD images (RGB+Depth) as input and exploits the depth information to improve its perception of the geometry (Section 3.4.5).

Concurrent with our work, the following studies also introduce a generalizable NeRF: RGBD-Net (Nguyen et al., 2021) builds a cost volume for the target view instead of source views, NeuralMVS(Rosu and Behnke, 2022) proposes a coarse to fine approach to increase speed, and NeuRay Liu et al. (2022) proposes a method to deal with occlusions.

3.3 Related Work

Multi-View Stereo. The purpose of Multi-View Stereo (MVS) is to estimate the dense representation of a scene given multiple overlapping images. This field has been extensively studied: first, with now called traditional methods (Kolmogorov and Zabih, 2002; De Bonet and Viola, 1999; Furukawa and Ponce, 2009; Schönberger et al., 2016) and more recently with methods relying on deep learning such as MVSNet (Yao et al., 2018), which outperformed the traditional ones. MVSNet estimates the depth from multiple views by extracting features from all images, aggregating them into a variance-based cost volume after warping each view onto the reference one, and finally, post-processing the cost volumes with a 3D-CNN. The memory needed to post-process the cost volume being the main bottleneck of MVSNet (Yao et al., 2018), R-MVSNet (Yao et al., 2019) proposed regularizing the cost volume along the depth direction with gated recurrent units while slightly sacrificing accuracy. To further reduce the memory impact, Gu et al. (2020), Cheng et al. (2020), and Yang et al. (2020) proposed cascaded architectures, where the cost volume is built at gradually finer scales, with the depth output computed in a coarse to fine manner without any compromise on the accuracy. Replacing the variance-based metric with group-wise correlation similarity is another approach to further decrease the memory usage of MVS networks (Xu and Tao, 2020). We found MVS architectures suitable for inferring the geometry and occlusions in a scene and conditioning a novel image renderer.

Novel View Synthesis. Early work on synthesizing novel views from a set of reference images was done by blending reference pixels according to specific weights (Debevec et al., 1996; Levoy and Hanrahan, 1996). The weights were computed according to ray-space proximity (Levoy and Hanrahan, 1996) or approximated geometry (Buehler et al., 2001; Debevec et al., 1996). To improve the computed geometry, some used the optical flow (Casas et al., 2015; Du et al., 2018) or soft blending (Penner and Zhang, 2017). Others synthesized a radiance field directly on a mesh (Debevec et al., 1998; Huang et al., 2020) or a point cloud (Aliev et al., 2020; Meshry et al., 2019). An advantage of these methods is that they can synthesize new views with a small number of references, but their performance is limited by the quality of 3D reconstruction (Jancosek and Pajdla, 2011; Schonberger and Frahm, 2016), and problems often arise in low-textured or reflective regions where stereo reconstruction tends to fail. Leveraging CNNs to predict volumetric representations stored in voxel grids (Kalantari et al., 2016; Penner and Zhang, 2017; Henzler et al., 2020) or Multi-Plane Images (Flynn et al., 2016; Zhou et al., 2018; Srinivasan et al., 2019; Flynn et al., 2019) produces photo-realistic renderings. Those methods rely on discrete volumetric representations of the scenes limiting their outputs' resolution. They also need to be trained on large datasets to store large numbers of samples resulting in extensive memory overhead.

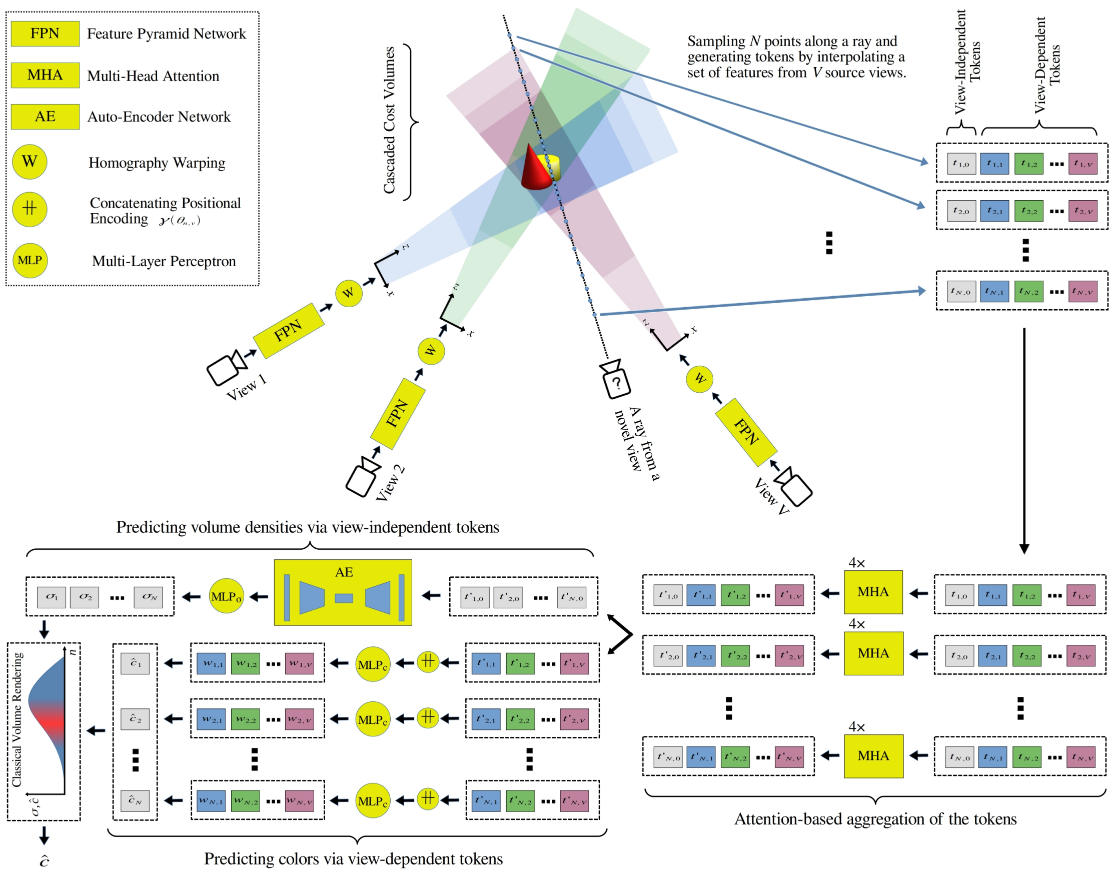
Neural Scene Representations. Recently, using neural networks to represent the geometry and appearance of scenes has allowed querying color and opacity in continuous space and viewing directions. NeRF (Mildenhall et al., 2020) achieves impressive results for novel view synthesis by optimizing a 5D neural radiance field for a scene. Building upon NeRF many improvements were made (Park et al., 2021a; Li et al., 2021b; Martin-Brualla et al., 2021; Peng et al., 2021; Schwarz et al., 2020; Srinivasan et al., 2021; Rockwell et al., 2021; DeVries et al., 2021; Barron et al., 2021), but the network needs to be optimized for hours or days for each new scene. Later works, such as GRF (Trevithick and Yang, 2021), pixelNeRF (Yu et al., 2021) and MINE (Li et al., 2021a), try to synthesize novel views with very sparse inputs, but their generalization ability to challenging scenes with complex specularities is highly restricted. MVSNeRF (Chen et al., 2021) proposes to use a low-resolution plane-swept cost volume to generalize rendering to new scenes with as few as three images without retraining. Once the cost volume is computed, MVSNeRF uses a 3D-CNN to aggregate image features. This 3D-CNN resembles the generic view interpolation function presented in IBRNet (Wang et al., 2021b) that allows rendering novel views on unseen scenes with few images. Inspired by MVSNeRF, our work first constructs cascaded cost volumes per source view and then aggregates the cost volumes of the views in an attention-based approach. The former allows for capturing high-resolution details, and the latter addresses occlusions.

3.4 Method

We use volume rendering techniques to synthesize novel views given a set of input source views. Our proposed architecture is presented in Figure 3.2, and the following sections provide details of our method.

3.4.1 Geometry Reasoner

Given a set of V nearby views $\{I_v\}_{v=1}^{V}$ with size $H \times W$, our geometry reasoner constructs cascaded cost volumes for each input view individually, following the same approach in CasMVSNet (Gu et al., 2020). First, each image goes through a Feature Pyramid Network



FPN Feature Pyramid Network
MHA Multi-Head Attention
AE Auto-Encoder Network
W Homography Warping
Concatenating Positional Encoding
MLP Multi-Layer Perceptron
Cascaded Cost Volumes
Sampling N points along a ray and generating tokens by interpolating a set of features from V source views.
View-Independent Tokens
View-Dependent Tokens
View 1
View 2
View V
A ray from a novel view
Predicting volume densities via view-independent tokens
Attention-based aggregation of the tokens
Predicting colors via view-dependent tokens
Classical Volume Rendering

Figure 3.2: The overview of GeoNeRF. 2D feature pyramids are first generated via Feature Pyramid Network (FPN) (Lin et al., 2017) for each source view v. We then construct cascaded cost volumes at three levels for each view by homography warping of its nearby views (see Section 3.4.1). Guided by the distribution of the cascaded cost volumes in the 3D space, $N = N_c + N_f$ points $\{x_n\}_{n=1}^{N}$ are sampled along a ray for a novel pose (see Section 3.4.2). By interpolating both 2D and 3D features ($\boldsymbol{f}_{n,v}^{(0)}$, $\{\boldsymbol{\Phi}_{n,v}^{(l)}\}_{l=0}^{2}$) from FPN and cascaded cost volumes for each sample point x_n, one view independent token $\boldsymbol{t}_{n,0}$ and V view-dependent tokens $\{\boldsymbol{t}_{n,v}\}_{v=1}^{V}$ are generated. These $V+1$ tokens go through four stacked Multi-Head Attention (MHA) layers and yield more refined tokens $\{\boldsymbol{t}'_{n,v}\}_{v=0}^{V}$. The MHA layers are shared among all sample points on a ray. Thereafter, the view-independent tokens $\{\boldsymbol{t}'_{n,0}\}_{n=1}^{N}$ are regularized and aggregated along the ray samples through the AE network, and volume densities $\{\boldsymbol{\sigma}_{\boldsymbol{n}}\}_{n=1}^{N}$ of the sampled points are estimated. Other tokens $\{\boldsymbol{t}'_{n,v}\}_{v=1}^{V}$, supplemented with the positional encodings $\{\gamma(\theta_{n,v})\}_{v=1}^{V}$, predict the color weights $\{\boldsymbol{w}_{n,v}\}_{v=1}^{V}$ with respect to source views, and the color $\hat{\boldsymbol{c}}_{\boldsymbol{n}}$ of each point is estimated in a weighted sum fashion (see Section 3.4.3). Finally, the color of the ray $\hat{\boldsymbol{c}}$ is rendered using classical volume rendering.

(FPN) (Lin et al., 2017) to generate semantic 2D features at three different scale levels.

$$\boldsymbol{f}_v^{(l)} = \mathrm{FPN}\,(I_v) \in \mathbb{R}^{\frac{H}{2^l} \times \frac{W}{2^l} \times 2^l C} \quad \forall\, l \in \{0,1,2\} \tag{3.1}$$

where FPN is the Feature Pyramid Network, C is the channel dimension at level 0, and l indicates the scale level. Once 2D features are generated, we follow the same approach in CasMVSNet to construct plane sweeps and cascaded cost volumes at three levels via differentiable homography warping. CasMVSNet originally estimates depth maps $\hat{\boldsymbol{D}}^{(\boldsymbol{l})}$ of the input images at three levels. The coarsest level ($l = 2$) consists of $D^{(2)}$ plane sweeps covering the whole depth range in the camera's frustum. Then, subsequent levels narrow the hypothesis range (decrease $D^{(l)}$) but increase the spatial resolution of each voxel by creating $D^{(l)}$ finer plane sweeps on both sides of the estimated depths from the previous level. As a result, the finer the cost volume is, the thinner the depth range it covers. We make two modifications to the CasMVSNet architecture and use it as the geometry reasoner. Firstly, we provide an additional output head to the network to produce multi-level semantic 3D features $\boldsymbol{\Phi}^{(\boldsymbol{l})}$ along with the estimated depth maps $\hat{\boldsymbol{D}}^{(\boldsymbol{l})}$. Secondly, we replace the variance-based metric in CasMVSNet with **group-wise correlation similarity** from Xu and Tao (2020) to construct lightweight volumes. Group-wise correlation decreases memory usage and inference time.

To be more specific, for each source view I_v, we first form a set of its nearby views Γ_v. Then, by constructing $D^{(l)}$ depth plane sweeps and homography warping techniques, we create multi-level cost volumes $\boldsymbol{P}_{\boldsymbol{v}}^{(\boldsymbol{l})}$ from 2D feature pyramids $\boldsymbol{f}^{(\boldsymbol{l})}$ of images in Γ_v using the group-wise correlation similarity metric from Xu and Tao (2020). We finally further process and regularize the cost volumes using 3D hourglass networks $R_{3D}^{(l)}$ and generate depth maps $\hat{\boldsymbol{D}}_{\boldsymbol{v}}^{(\boldsymbol{l})} \in \mathbb{R}^{\frac{H}{2^l} \times \frac{W}{2^l} \times 1}$ and 3D feature maps $\boldsymbol{\Phi}_{\boldsymbol{v}}^{(\boldsymbol{l})} \in \mathbb{R}^{D^{(l)} \times \frac{H}{2^l} \times \frac{W}{2^l} \times C}$:

$$\hat{\boldsymbol{D}}_{\boldsymbol{v}}^{(\boldsymbol{l})}, \boldsymbol{\Phi}_{\boldsymbol{v}}^{(\boldsymbol{l})} = R_{3D}^{(l)}\left(\boldsymbol{P}_{\boldsymbol{v}}^{(\boldsymbol{l})}\right) \quad \forall l \in \{0, 1, 2\} \tag{3.2}$$

3.4.2 Sampling Points on a Novel Ray

Once the features from the geometry reasoner are generated, we render novel views with the ray-casting approach. For each camera ray at a novel camera pose, we first sample N_c points along the ray uniformly to cover the whole depth range. Furthermore, we estimate a stepwise probability density function $p_0(x)$ along the ray representing the probability that a point x is covered by a full-resolution partial cost volume $\boldsymbol{P}^{(\boldsymbol{0})}$. Voxels inside the thinnest, full-resolution cost volumes $\{\boldsymbol{P}_{\boldsymbol{v}}^{(\boldsymbol{0})}\}_{v=1}^{V}$ contain the most valuable information about the surfaces and geometry. Therefore, we sample N_f more points from $p_0(x)$ distribution. Unlike previous works(Mildenhall et al., 2020; Yu et al., 2021; Wang et al., 2021b; Reizenstein et al., 2021; Arandjelović and Zisserman, 2021) that require training two networks simultaneously (one coarse and one fine network) and rendering volume densities from the coarse network to resample more points for the fine one, we sample a mixture of $N = N_c + N_f$ valuable points before rendering the ray without any computation overhead or network duplication thanks to the design of our geometry reasoner.

3.4.3 Renderer

For all sample points $\{x_n\}_{n=1}^N$, we interpolate the full-resolution 2D features $\boldsymbol{f}_{\boldsymbol{n},\boldsymbol{v}}^{(0)}$ and the three-level 3D features $\{\boldsymbol{\Phi}_{\boldsymbol{n},\boldsymbol{v}}^{(\boldsymbol{l})}\}_{l=0}^2$ from all source views. We also define an occlusion mask $M_{n,v}$ for each point x_n with respect to each view v. Formally, if a point x_n stands behind the estimated full-resolution depth map $\hat{\boldsymbol{D}}_{\boldsymbol{v}}^{(0)}$ (being occluded) or the projection of x_n to the camera plane of view v lies outside of the image plane (being outside of the camera frustum), we set $M_{n,v} = 0$ and discard view v from the rendering process of point x_n. Next, we create a view-independent token $t_{n,0}$ and V view-dependent tokens $\{t_{n,v}\}_{v=1}^V$ by utilizing the interpolated features for each point x_n:

$$\begin{aligned} \boldsymbol{t}_{\boldsymbol{n},\boldsymbol{v}} &= \mathrm{LT}\left(\left[\boldsymbol{f}_{\boldsymbol{n},\boldsymbol{v}}^{(0)}; \{\boldsymbol{\Phi}_{\boldsymbol{n},\boldsymbol{v}}^{(\boldsymbol{l})}\}_{l=0}^2\right]\right) \quad \forall v \in \{1,\ldots,V\} \\ \boldsymbol{t}_{\boldsymbol{n},\boldsymbol{0}} &= \mathrm{LT}\left(\left[mean\{\boldsymbol{f}_{\boldsymbol{n},\boldsymbol{v}}^{(0)}\}_{v=1}^V; var\{\boldsymbol{f}_{\boldsymbol{n},\boldsymbol{v}}^{(0)}\}_{v=1}^V\right]\right) \end{aligned} \tag{3.3}$$

where $\mathrm{LT}(\cdot)$ and $[\cdot\ ;\ \cdot]$ denote respectively linear transformation and concatenation. $\boldsymbol{t}_{\boldsymbol{n},\boldsymbol{0}}$ could be considered as a global understanding of the scene at point x_n, while $\boldsymbol{t}_{\boldsymbol{n},\boldsymbol{v}}$ represents the understanding of the scene from source view v. The global and view-dependent tokens are aggregated through four stacked Multi-Head Attention (MHA) layers, which are introduced in Transformers (Dosovitskiy et al., 2021; Vaswani et al., 2017):

$$\{\boldsymbol{t}'_{\boldsymbol{n},\boldsymbol{v}}\}_{v=0}^V = {}^{4\times}\mathrm{MHA}\left(\boldsymbol{t}_{\boldsymbol{n},\boldsymbol{0}}, \{\boldsymbol{t}_{\boldsymbol{n},\boldsymbol{v}}, M_{n,v}\}_{v=1}^V\right) \tag{3.4}$$

Our MHA layers also take the occlusion masks $M_{n,v}$ as inputs and force the occluded views' attention scores to zero to prevent them from contributing to the aggregation.

The global view-independent output tokens $\{\boldsymbol{t}'_{\boldsymbol{n},\boldsymbol{0}}\}_{n=1}^N$ now have access to all necessary data to learn the geometry of the scene and estimate volume densities. We further regularize these tokens through an auto-encoder-style (AE) network in the ray dimension (n). The AE network learns the global geometry along the ray via convolutional layers and predicts more coherent volume densities $\boldsymbol{\sigma}_{\boldsymbol{n}}$:

$$\{\boldsymbol{\sigma}_{\boldsymbol{n}}\}_{n=1}^N = \mathrm{MLP}_\sigma\left(\mathrm{AE}\left(\{\boldsymbol{t}'_{\boldsymbol{n},\boldsymbol{0}}\}_{n=1}^N\right)\right) \tag{3.5}$$

where MLP_σ is a simple two-layer perceptron. We argue that convolutionally processing the tokens with the AE network along the ray dimension (n) is a proper inductive bias and significantly reduces the computation resources compared to methods like IBRNet (Wang et al., 2021b) and NeRFormer (Reizenstein et al., 2021), which employ an attention-based architecture because the geometry of a scene is naturally continuous, and accordingly, closer points are more likely related.

View-dependent tokens $\{\boldsymbol{t}'_{\boldsymbol{n},\boldsymbol{v}}\}_{v=1}^V$, together with two additional inputs, are used for color prediction. We project each point x_n to source views' image planes and interpolate the color samples $c_{n,v}$. We also calculate the angle between the novel camera ray and the line that passes through the camera center of the source view v and x_n. This angle $\theta_{n,v}$ represents the similarity between the camera pose of the source view v and the novel view. Each point's color is estimated via a weighted sum of the non-occluded views' colors:

$$\begin{aligned}\boldsymbol{w}_{\boldsymbol{n},\boldsymbol{v}} &= \mathrm{Softmax}\left(\left\{\mathrm{MLP}_c\left([\boldsymbol{t}'_{\boldsymbol{n},\boldsymbol{v}};\gamma(\theta_{n,v})]\right), M_{n,v}\right\}_{v=1}^{V}\right)\\ \hat{\boldsymbol{c}}_{\boldsymbol{n}} &= \sum_{v=1}^{V} \boldsymbol{w}_{\boldsymbol{n},\boldsymbol{v}} c_{n,v} \quad \forall n \in \{1,2,\dots,N\}\end{aligned} \tag{3.6}$$

where $\gamma(\cdot)$ is the sinusoidal positional encoding proposed in NeRF (Mildenhall et al., 2020), and MLP_c is a simple two-layer perceptron. The Softmax function also takes the occlusion masks $M_{n,v}$ as input to exclude occluded views.

Once volume densities and colors are predicted, our model renders, as in NeRF (Mildenhall et al., 2020), the color of the camera ray at a novel pose using the volume rendering approach:

$$\hat{\boldsymbol{c}} = \sum_{n=1}^{N} \exp\left(-\sum_{k=1}^{n-1} \boldsymbol{\sigma}_{\boldsymbol{k}}\right)\left(1-\exp\left(-\boldsymbol{\sigma}_{\boldsymbol{n}}\right)\right)\hat{\boldsymbol{c}}_{\boldsymbol{n}} \tag{3.7}$$

In addition to the rendered color, our model also outputs the estimated depth $\hat{\boldsymbol{d}}$ for each ray:

$$\hat{\boldsymbol{d}} = \sum_{n=1}^{N} \exp\left(-\sum_{k=1}^{n-1} \boldsymbol{\sigma}_{\boldsymbol{k}}\right)\left(1-\exp\left(-\boldsymbol{\sigma}_{\boldsymbol{n}}\right)\right) z_n \tag{3.8}$$

where z_n is the depth of point x_n with respect to the novel pose. This auxiliary output is helpful for training and supervising our generalizable model (see Section 3.4.4).

3.4.4 Loss Functions

The primary loss function when we train our generalizable model on various scenes and the **only** loss when we fine-tune it on a specific scene is the mean squared error between the rendered colors and ground truth pixel colors:

$$\mathscr{L}_c = \frac{1}{|R|}\sum_{r\in R}\left\|\hat{\boldsymbol{c}}(r) - c_{gt}(r)\right\|^2 \tag{3.9}$$

where R is the set of rays in each training batch and c_{gt} is the ground truth color.

DS-NeRF (Deng et al., 2022) shows that depth supervision can help NeRF train faster with fewer input views. Moreover, numerous works (Kaya et al., 2022; Oechsle et al., 2021; Wang et al., 2021a; Yariv et al., 2021) show that despite the high-quality color rendering, NeRF has difficulty reconstructing 3D geometry and surface normals. Accordingly, for training samples coming from datasets with ground truth depths, we also output the predicted depth $\hat{\boldsymbol{d}}$ for each ray and supervise it if the ground truth depth of that pixel is available:

$$\mathscr{L}_d = \frac{1}{|R_d|}\sum_{r\in R_d}\left\|\hat{\boldsymbol{d}}(r) - d_{gt}(r)\right\|_{s1} \tag{3.10}$$

where R_d is the set of rays from samples with ground truth depths and d_{gt} is the pixel ground truth depth and $||\cdot||_{s1}$ is the smooth L_1 loss.

Lastly, we supervise cascaded depth estimation networks in our geometry reasoner. For datasets with ground truth depth, the loss is defined as:

$$\mathcal{L}_D^{(l)} = \frac{2^{-l}}{|V|} \sum_{v=1}^{V} \langle \left\| \hat{\boldsymbol{D}}_{\boldsymbol{v}}^{(\boldsymbol{l})} - D_v^{(l)} \right\|_{s1} \rangle \tag{3.11}$$

where $D_v^{(l)}$ is the ground truth depth map of view v resized to scale level l, and $< \cdot >$ denotes averaging over all pixels. For training samples without ground truth depths, we self-supervise the depth maps. We take the rendered ray depths as pseudo-ground truth and warp their corresponding colors and estimated depths from all source views using camera transformation matrices. If the ground truth pixel color of a ray is consistent with the warped color of a source view, and it is located in a textured neighborhood, we allow $\hat{\boldsymbol{d}}$ to supervise the geometry reasoner for that view. Formally:

$$\begin{aligned}
&\mathcal{L}_D^{(l)} = \frac{2^{-l}}{|V||R|} \sum_{v=1}^{V} \sum_{r \in R} M_v(r) \left\| \hat{\boldsymbol{D}}_{\boldsymbol{v}}^{(\boldsymbol{l})}(r_v) - \hat{\boldsymbol{d}}(r_v) \right\|_{s1} \\
&\text{where} \quad r_v = T_{\to v}\left(r, \hat{\boldsymbol{d}}(r)\right) \\
&\text{and} \quad M_v(r) = \begin{cases} 1 & \text{if } \left| I_v(r_v) - c_{gt}(r) \right| < \epsilon_c \text{ and } V_5\left(I_v(r_v)\right) > \epsilon_t \\ 0 & \text{otherwise} \end{cases}
\end{aligned} \tag{3.12}$$

Given a ray r at a novel pose with rendered depth $\hat{\boldsymbol{d}}(r)$, $T_{\to v}\left(r, \hat{\boldsymbol{d}}(r)\right)$ transforms the ray to its correspondent ray from source view v using camera matrices. $\hat{\boldsymbol{d}}(r_v)$ denotes the rendered depth of the correspondent ray with respect to source view v, and $M_v(r)$ validates the texturedness and color consistency. We keep pixels whose variance $V_5(\cdot)$ in their 5×5 pixels neighborhood is higher than ϵ_t, and whose color differs less than ϵ_c from the color of the ray r. The aggregated loss function for our generalizable model is:

$$\mathcal{L} = \mathcal{L}_c + 0.1\mathcal{L}_d + \lambda \sum_{l=0}^{2} \mathcal{L}_D^{(l)} \tag{3.13}$$

where λ is 1.0 if the supervision is with ground truth depths and is 0.1 if it is with pseudo-ground truth rendered depths. For **fine-tuning** on a single scene, regardless of the availability of depth data, we only use $\mathcal{L}_c$ as the loss function.

3.4.5 Compatibility with RGBD data

Concerning the ubiquitousness of the embedded depth sensors in devices nowadays, we also propose an RGBD compatible model, GeoNeRF$_{+\text{D}}$, by making a small modification to the geometry reasoner. We assume an **incomplete**, low-resolution, noisy depth map $D_v \in \mathbb{R}^{\frac{H}{4} \times \frac{W}{4} \times 1}$

is available for each source view v. When we construct the coarsest cost volume $\boldsymbol{P}_{\boldsymbol{v}}^{\mathbf{(2)}}$ with $D^{(2)}$ depth planes, we also construct a binary volume $B_v \in \mathbb{R}^{D^{(2)} \times \frac{H}{4} \times \frac{W}{4} \times 1}$ and concatenate it with $\boldsymbol{P}_{\boldsymbol{v}}^{\mathbf{(2)}}$ before feeding them to the $R_{3D}^{(2)}$ network:

$$B_v(d,h,w) = \begin{cases} 1 & \text{if } Q\left(D_v\left(h,w\right)\right) \equiv d \\ 0 & \text{otherwise} \end{cases} \tag{3.14}$$

where $Q(\cdot)$ maps and quantizes real depth values to the depth plane indices. B_v plays the role of coarse guidance of the geometry in GeoNeRF$_{\text{+D}}$. As a result of this design, the model is robust to the quality and sparsity of the depth inputs.

3.5 Experiments

Training datasets. We train our model on the real DTU dataset (Jensen et al., 2014) and real forward-facing datasets from LLFF (Mildenhall et al., 2019) and IBRNet (Wang et al., 2021b). We exclude views with incorrect exposure from the DTU dataset (Jensen et al., 2014) following the practice in pixelNeRF (Yu et al., 2021) and use the same 88 scenes for training as in pixelNeRF (Yu et al., 2021) and MVSNeRF (Chen et al., 2021). Ground truth depths of DTU (Jensen et al., 2014), provided by Yao et al. (2018), are the only data that is directly used for depth supervision. For samples from forward-facing datasets (35 scenes from LLFF (Mildenhall et al., 2019) and 67 scenes from IBRNet (Wang et al., 2021b)), depth supervision is in the self-supervised form.

Evaluation datasets. We evaluate our model on the 16 test scenes of DTU MVS (Jensen et al., 2014), the 8 test scenes of real forward-facing dataset from LLFF (Mildenhall et al., 2019), and the 8 scenes in NeRF realistic synthetic dataset (Mildenhall et al., 2020). We follow the same evaluation protocol in NeRF (Mildenhall et al., 2020) for the NeRF synthetic dataset (Mildenhall et al., 2020) and LLFF dataset (Mildenhall et al., 2019) and the same protocol in MVSNeRF (Chen et al., 2021) for the DTU dataset (Jensen et al., 2014). Specifically, for LLFF (Mildenhall et al., 2019), we hold out $\frac{1}{8}$ of the views of the unseen scenes, and for DTU (Jensen et al., 2014), we hold out 4 views of the unseen scenes for testing and leave the rest for fine-tuning.

Implementation details. We train the generalizable GeoNeRF for 250k iterations. For each iteration, one scene is randomly sampled, and 512 rays are randomly selected as the training batch. For training on a specific scene, we only fine-tune the model for 10k iterations (GeoNeRF$_{\text{10k}}$), in contrast to NeRF (Mildenhall et al., 2020) that requires 200k–500k optimization steps per scene. Since our renderer's architecture is agnostic to the number of source views, we flexibly employ a different number of source views V for training and evaluation to reduce memory usage. We use $V = 6$ source views for training the generalizable model and

DTU MVS Dataset (Jensen et al., 2014) [†]				
Method	Settings	PSNR↑	SSIM↑	LPIPS↓
pixelNeRF (Yu et al., 2021)	No per-scene Optimization	19.31	0.789	0.382
IBRNet (Wang et al., 2021b)		26.04	0.917	0.190
MVSNeRF (Chen et al., 2021)		<u>26.63</u>	<u>0.931</u>	<u>0.168</u>
GeoNeRF		**31.34**	**0.959**	**0.060**
IBRNet (Wang et al., 2021b)	Per-scene Optimization	31.35	0.956	<u>0.131</u>
MVSNeRF (Chen et al., 2021)		28.50	0.933	0.179
NeRF (Mildenhall et al., 2020)		27.01	0.902	0.263
GeoNeRF$_{10k}$		**31.66**	**0.961**	**0.059**
GeoNeRF$_{1k}$		<u>31.52</u>	<u>0.960</u>	**0.059**

[†] The evaluations of the baselines on the DTU MVS dataset (Jensen et al., 2014) are borrowed from the paper MVSNeRF (Chen et al., 2021). Also, metrics are calculated for all methods on this dataset on foreground pixels, whose ground truth depths stand inside the scene bound.

Table 3.1: Quantitative comparison of our proposed GeoNeRF with existing generalizable NeRF models in terms of PSNR (higher is better), SSIM (Wang et al., 2004) (higher is better), and LPIPS (Zhang et al., 2018a) (lower is better) metrics on the DTU MVS Dataset (Jensen et al., 2014). Highlights are **best** and <u>second best</u>. GeoNeRF is superior to the existing approaches in the experiments in which the methods are evaluated without any per-scene optimization (the top row). Notably, GeoNeRF outperforms others with a significant margin on this dataset which has relatively sparser source views. The bottom row of the table presents the evaluation of the methods when they are fine-tuned on each scene separately, as well as a comparison with vanilla NeRF (Mildenhall et al., 2020), which is per-scene optimized. After fine-tuning for only 10k iterations, our GeoNeRF$_{10k}$ produces competitive results with NeRF. Remarkably, even after fine-tuning for 1k iterations (approximately one hour on a single V100 GPU), GeoNeRF$_{1k}$ reaches 99.81% of the GeoNeRF$_{10k}$'s performance on average, which is another evidence for efficient convergence of our model on novel scenes.

$V = 9$ for evaluation. For fine-tuning, we select V based on the images' resolution and available GPU memory. Specifically, we set $V = 9$ for the DTU dataset Jensen et al. (2014) and $V = 7$ for the other two datasets. We fix the number of sample points on a ray to $N_c = 96$ and $N_f = 32$ for all scenes. We utilize Adam (Kingma and Ba, 2014) as the optimizer with a learning rate of 5×10^{-4} for training the generalizable model and a learning rate of 2×10^{-4} for fine-tuning. A cosine learning rate scheduler (Loshchilov and Hutter, 2017) without restart is also applied to the optimizer. For the details of our networks' architectures, refer to Section 3.7.1.

3.5.1 Experimental Results

We evaluate our model and provide a comparison with the original vanilla NeRF (Mildenhall et al., 2020) and the existing generalizable NeRF models: pixelNeRF (Yu et al., 2021), IBR-Net (Wang et al., 2021b), and MVSNeRF (Chen et al., 2021). The authors of NeRFormer (Reizen-

NeRF Realistic Synthetic Dataset (Mildenhall et al., 2020)				
Method	Settings	PSNR↑	SSIM↑	LPIPS↓
pixelNeRF (Yu et al., 2021)	No per-scene Optimization	7.39	0.658	0.411
IBRNet (Wang et al., 2021b)		<u>25.49</u>	<u>0.916</u>	<u>0.100</u>
MVSNeRF (Chen et al., 2021)		23.62	0.897	0.176
GeoNeRF		**28.33**	**0.938**	**0.087**
IBRNet (Wang et al., 2021b)	Per-scene Optimization	28.14	0.942	0.072
MVSNeRF (Chen et al., 2021) †		27.07	0.931	0.168
NeRF (Mildenhall et al., 2020)		**31.01**	0.947	0.081
GeoNeRF$_{10k}$		<u>30.42</u>	**0.956**	**0.055**
GeoNeRF$_{1k}$		29.83	<u>0.952</u>	<u>0.061</u>

† For fine-tuning, MVSNeRF (Chen et al., 2021) discards its CNNs and directly optimizes 3D features. Direct optimization without regularization severely suffers from overfitting which is not reflected in their reported accuracy because their evaluation is done on a custom test set instead of the standard one. *E.g.*, their PSNR on NeRF dataset (Mildenhall et al., 2020) would drop from **27.07** to **20.02** if it was evaluated on the standard test set.

Table 3.2: Quantitative comparison of our proposed GeoNeRF with existing generalizable NeRF models in terms of PSNR (higher is better), SSIM (Wang et al., 2004) (higher is better), and LPIPS (Zhang et al., 2018a) (lower is better) metrics on the NeRF realistic synthetic Dataset (Mildenhall et al., 2020). Highlights are **best** and <u>second best</u>. GeoNeRF is superior to the existing approaches in the experiments in which the methods are evaluated without any per-scene optimization (the top row). Notably, GeoNeRF outperforms others with a significant margin on this dataset which has relatively sparser source views. It is also worth noting that our method generalizes outstandingly well on the NeRF synthetic dataset (Mildenhall et al., 2020) although our training dataset only contains real scenes that greatly differ from the synthetic scenes. The bottom row of the table presents the evaluation of the methods when they are fine-tuned on each scene separately, as well as a comparison with vanilla NeRF (Mildenhall et al., 2020), which is per-scene optimized for 500k iterations. After fine-tuning for only 10k iterations, our GeoNeRF$_{10k}$ produces competitive results with NeRF. Remarkably, even after fine-tuning for 1k iterations (approximately one hour on a single V100 GPU), GeoNeRF$_{1k}$ reaches 95.91% of the GeoNeRF$_{10k}$'s performance on average, which is another evidence for efficient convergence of our model on novel scenes.

stein et al., 2021) did not publish their code, did not benchmark their method on NeRF benchmarking datasets, nor test state-of-the-art generalizable NeRF models on their own dataset. They perform on par with NeRF in their experiments with scene-specific optimization and train and test their generalizable model on specific object categories separately. A quantitative comparison is provided in Tables 3.1, 3.2, and 3.3 in terms of PSNR, SSIM (Wang et al., 2004), and LPIPS (Zhang et al., 2018a). The results show the superiority of our GeoNeRF model with respect to the previous generalizable models. Moreover, when fine-tuned on the scenes for only 10k iterations, GeoNeRF$_{10k}$ produces competitive results with NeRF, which requires lengthy per-scene optimization. We further show that even after 1k iterations (approximately

Real Forward Facing (Mildenhall et al., 2019)				
Method	Settings	PSNR↑	SSIM↑	LPIPS↓
pixelNeRF (Yu et al., 2021)	No per-scene Optimization	11.24	0.486	0.671
IBRNet (Wang et al., 2021b)		<u>25.13</u>	<u>0.817</u>	<u>0.205</u>
MVSNeRF (Chen et al., 2021)		21.93	0.795	0.252
GeoNeRF		**25.44**	**0.839**	**0.180**
IBRNet (Wang et al., 2021b)	Per-scene Optimization	**26.73**	0.851	0.175
MVSNeRF (Chen et al., 2021)		25.45	**0.877**	0.192
NeRF (Mildenhall et al., 2020)		26.50	0.811	0.250
GeoNeRF$_{10k}$		<u>26.58</u>	<u>0.856</u>	**0.162**
GeoNeRF$_{1k}$		26.31	0.852	<u>0.164</u>

Table 3.3: Quantitative comparison of our proposed GeoNeRF with existing generalizable NeRF models in terms of PSNR (higher is better), SSIM (Wang et al., 2004) (higher is better), and LPIPS (Zhang et al., 2018a) (lower is better) metrics on the Real Forward Facing (Mildenhall et al., 2019). Highlights are **best** and <u>second best</u>. GeoNeRF is superior to the existing approaches in all experiments in which the methods are evaluated without any per-scene optimization (the top row). The bottom row of the table presents the evaluation of the methods when they are fine-tuned on each scene separately, as well as a comparison with vanilla NeRF (Mildenhall et al., 2020), which is per-scene optimized for 200k iterations. After fine-tuning for only 10k iterations, our GeoNeRF$_{10k}$ produces competitive results with NeRF. Remarkably, even after fine-tuning for 1k iterations (approximately one hour on a single V100 GPU), GeoNeRF$_{1k}$ reaches 99.10% of the GeoNeRF$_{10k}$'s performance on average, which is another evidence for efficient convergence of our model on novel scenes.

one hour on a single V100 GPU), GeoNeRF$_{1k}$'s results are comparable with NeRF's.

The qualitative comparisons of our model with existing methods on different datasets are provided in Figures 3.3 and 3.4. The images produced by our GeoNeRF model better preserve the details of the scene and contain fewer artifacts. For further qualitative analysis, an extensive ablation study, and limitations of our model, refer to the supplementary material in Section 3.7.

3.5.2 Sensitivity to Source Views

We conducted two experiments to investigate the robustness of our model to the number and quality of input source views. We first evaluated the impact of the number of source views on our model in Table 3.4. The results demonstrate the robustness of our method to the sparsity of source views and suggest that GeoNeRF produces high-quality images even with a lower number of source images. Furthermore, we show that our method can operate with both close and distant source views. Table 3.5 shows the performance when we discard K nearest views to the novel pose and use the remaining source views for rendering. While distant source views are naturally less informative and degrade the quality, our model does not incur a significant decrease in performance.

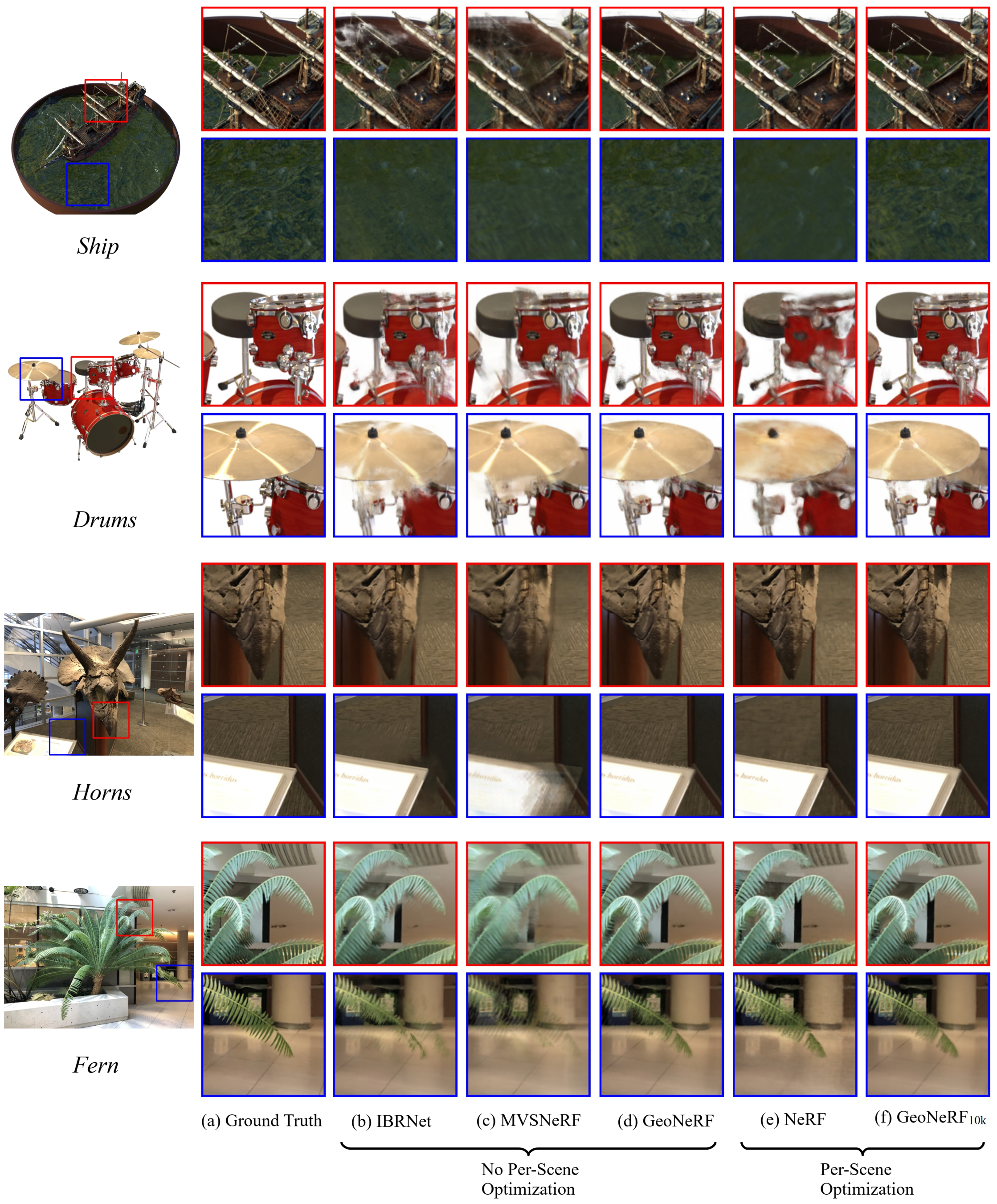


Figure 3.3: Qualitative comparison of the methods on the NeRF synthetic dataset (Martin-Brualla et al., 2021) (*Ship* and *Drums*) and the real forward-facing dataset (Mildenhall et al., 2019) (*Horns* and *Fern*). Our proposed GeoNeRF more accurately preserves the details of the scenes while it generates fewer artifacts than IBRNet (Wang et al., 2021b) and MVSNeRF (Chen et al., 2021) (*e.g.*, the leaves in *Fern* or the cymbal in *Drums*). After fine-tuning our model only for 10k iterations on each individual scene (GeoNeRF$_{10k}$), the results are competitive with per-scene optimized vanilla NeRF (Mildenhall et al., 2020). Compared with NeRF, GeoNeRF models produce smoother surfaces in *Drums* and higher quality textures for the water in *Ship* and for the floor in *Horns*.

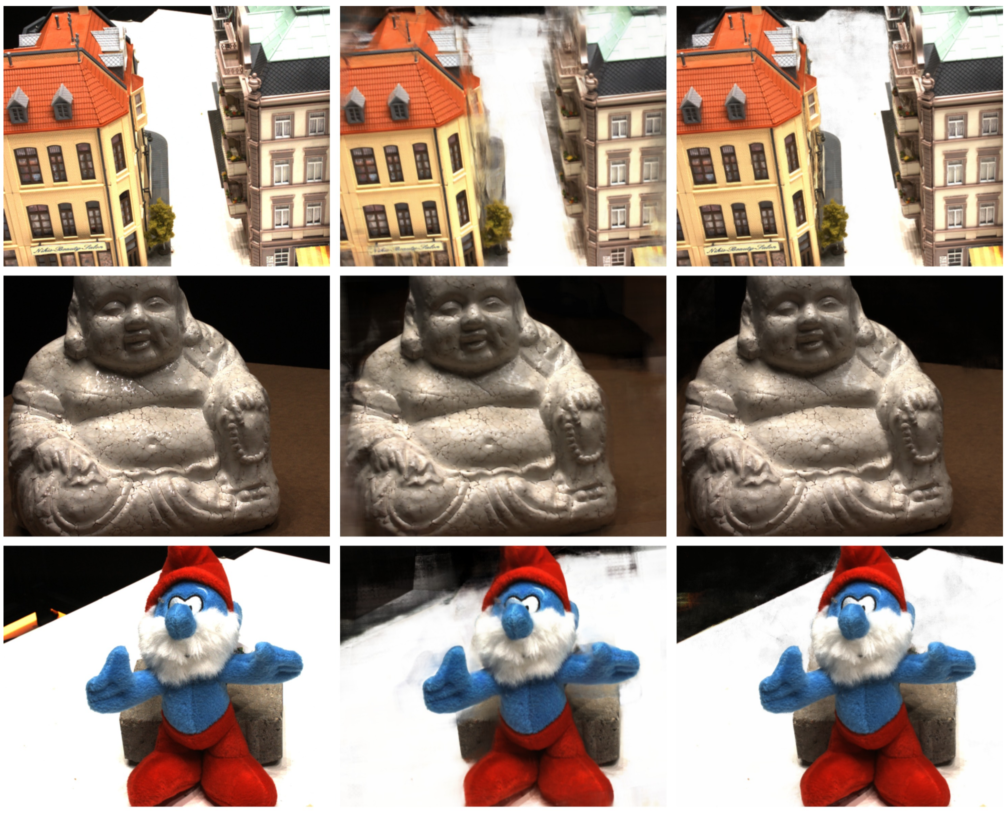


Figure 3.4: Qualitative comparison of our generalizable GeoNeRF model with MVSNeRF (Chen et al., 2021), the state-of-the-art model on the DTU dataset (Jensen et al., 2014). Images are from DTU test scenes. Our method renders sharper images with fewer artifacts.

3.5.3 Results with RGBD Images

To evaluate our RGBD compatible model, GeoNeRF$_{+D}$, we use the DTU dataset (Jensen et al., 2014) to mimic the real scenario where incomplete, low-resolution depth images accompany RGB images. We feed our model the DTU images with a resolution of 600×800, while we resize their incomplete depths to 150×200. The comparison of the performance of GeoNeRF, with and without depth inputs is presented in Table 3.6. The results confirm that our GeoNeRF$_{+D}$ model adapts to RGBD images and renders higher-quality outputs.

3.6 Conclusion

We proposed GeoNeRF, a generalizable learning-based novel view synthesis method that renders state-of-the-art quality images for complex scenes without per-scene optimization. Our method leverages the recent architectures in the multi-view stereo field to understand the

Number of source views	PSNR↑	SSIM↑	LPIPS↓
3	24.33	0.794	0.212
4	25.05	0.823	0.183
5	25.25	0.832	0.178
6	25.37	0.837	**0.176**
7	25.37	0.838	0.177
8	25.39	0.838	0.179
9	**25.44**	**0.839**	0.180
IBRNet (uses 10 views)	25.13	0.817	0.205

Table 3.4: Quantitative analysis of the robustness of our GeoNeRF to the number of input source views on the LLFF (Mildenhall et al., 2019) test scenes, besides a comparison with IBRNet (Wang et al., 2021b), which uses 10 source views.

K:	0	2	4	6	8
PSNR↑	**25.44**	24.18	23.35	22.74	22.06
SSIM↑	**0.839**	0.813	0.791	0.770	0.747
LPIPS↓	**0.180**	0.212	0.235	0.253	0.276

Table 3.5: Quantitative analysis of the sensitivity of our GeoNeRF to discarded first K nearest neighbors on the LLFF (Mildenhall et al., 2019) test scenes.

Model	PSNR↑	SSIM↑	LPIPS↓
GeoNeRF	31.34	0.959	0.060
GeoNeRF$_{+D}$	**31.58**	**0.961**	**0.057**

Table 3.6: A comparison of the performance of our RGBD compatible GeoNeRF$_{+D}$ and original GeoNeRF on DTU (Jensen et al., 2014) test scenes. For the details of this experiment, see Section 3.5.3.

scene's geometry and occlusions by constructing cascaded cost volumes for source views. The data coming from the source views are then aggregated through an attention-based network, and images for novel poses are synthesized while are conditioned on these data. An advanced algorithm to select a proper set of nearby views or an adaptive approximation of the optimal number of required cost volumes for a scene could be promising extensions to our method.

Input	Layer	Output
Input	ConvBnReLU(3, 8, 3, 1)	conv0_0
conv0_0	ConvBnReLU(8, 8, 3, 1)	conv0
conv0	ConvBnReLU(8, 16, 5, 2)	conv1_0
conv1_0	ConvBnReLU(16, 16, 3, 1)	conv1_1
conv1_1	ConvBnReLU(16, 16, 3, 1)	conv1
conv1	ConvBnReLU(16, 32, 5, 2)	conv2_0
conv2_0	ConvBnReLU(32, 32, 3, 1)	conv2_1
conv2_1	ConvBnReLU(32, 32, 3, 1)	conv2
conv2	Conv(32, 32, 1, 1)	feat2
conv1	Conv(16, 32, 1, 1)	f1_0
conv0	Conv(8, 32, 1, 1)	f0_0
(feat2, f1_0)	Upsample_and_Add(x, y)	f1_1
(f1_1, f0_0)	Upsample_and_Add(x, y)	f0_1
f1_1	Conv(32, 16, 3, 1)	feat1
f0_1	Conv(32, 8, 3, 1)	feat0

Table 3.7: Network architecture of Feature Pyramid Network (FPN), where *feat2*, *feat1*, and *feat0* are output feature pyramids. Conv(c_{in}, c_{out}, k, s) stands for a 2D convolution with input channels c_{in}, output channels c_{out}, kernel size of k, and stride of s. ConvBnReLU represents a Conv layer followed by Batch Normalization and ReLU nonlinearity. Upsample_and_Add(x, y) adds y to the bilinearly upsampled of x.

3.7 Supplementary Material

3.7.1 Additional Technical Details

As stated in Section 3.4.1, we borrow the architecture of our geometry reasoner from Cas-MVSNet (Gu et al., 2020). We construct $D^{(2)} = 48$ depth planes for the coarsest cost volume, $D^{(1)} = 32$ for the intermediate one, and $D^{(0)} = 8$ for the finest full-resolution cost volume. We use channel size $C = 8$ in group-wise correlation similarity calculations. When training the generalizable model, we create a set of 3–5 nearby source views for constructing each cost volume, whereas for fine-tuning and evaluating, we always use a set of 5 nearby views. Also, we scale input images with a factor uniformly sampled from {1.0, 0.75, 0.5} when we train our generalizable model.

The network architectures of Feature Pyramid Network (FPN), 3D regularizer ($R^{(l)}_{3D}$), and the auto-encoder (AE) are provided in Tables 3.7, 3.8, and 3.9 respectively.

Input	Layer	Output
Input	ConvBnReLU(8, 8, 3, 1)	conv0
conv0	ConvBnReLU(8, 16, 3, 2)	conv1
conv1	ConvBnReLU(16, 16, 3, 1)	conv2
conv2	ConvBnReLU(16, 32, 3, 2)	conv3
conv3	ConvBnReLU(32, 32, 3, 1)	conv4
conv4	ConvBnReLU(32, 64, 3, 2)	conv5
conv5	ConvBnReLU(64, 64, 3, 1)	conv6
conv6	TrpsConvBnReLU(64, 32, 3, 2)	x_0
(conv4, x_0)	Add(x, y)	x_1
x_1	TrpsConvBnReLU(32, 16, 3, 2)	x_2
(conv2, x_2)	Add(x, y)	x_3
x_3	TrpsConvBnReLU(16, 8, 3, 2)	x_4
(conv0, x_4)	Add(x, y)	x_5
x_5	ConvBnReLU(8, 8, 3, 1)	prob_0
prob_0	Conv(8, 1, 3, 1)	prob
x_5	ConvBnReLU(8, 8, 3, 1)	feat

Table 3.8: Network architecture of the 3D regularizer ($R_{3D}^{(l)}$), where *feat* is the output 3D feature map $\boldsymbol{\Phi}^{(l)}$ and *prob* is the output probability which is used to regress the depth map $\hat{\boldsymbol{D}}^{(l)}$. Conv($c_{in}$, c_{out}, k, s) stands for a 3D convolution with input channels c_{in}, output channels c_{out}, kernel size of k, and stride of s. ConvBnReLU represents a Conv layer followed by Batch Normalization and ReLU nonlinearity, and TrpsConv stands for transposed 3D convolution. Add(x, y) simply adds y to x.

Input	Layer	Output
Input	ConvLnELU(32, 64, 3, 1)	conv1_0
conv1_0	MaxPool	conv1
conv1	ConvLnELU(64, 128, 3, 1)	conv2_0
conv2_0	MaxPool	conv2
conv2	ConvLnELU(128, 128, 3, 1)	conv3_0
conv3_0	MaxPool	conv3
conv3	TrpsConvLnELU(128, 128, 4, 2)	x_0
[conv2 ; x_0]	TrpsConvLnELU(256, 64, 4, 2)	x_1
[conv1 ; x_1]	TrpsConvLnELU(128, 32, 4, 2)	x_2
[Input ; x_2]	ConvLnELU(64, 32, 3, 1)	Output

Table 3.9: Network architecture of the auto-encoder network (AE). Conv(c_{in}, c_{out}, k, s) stands for a 1D convolution with input channels c_{in}, output channels c_{out}, kernel size of k, and stride of s. ConvLnELU represents a Conv layer followed by Layer Normalization and ELU nonlinearity, and TrpsConv stands for transposed 1D convolution. MaxPool is a 1D max pooling layer with a stride of 2, and $[\cdot\,;\cdot]$ denotes concatenation.

3.7.2 Additional Qualitative Analysis

Full-size examples of rendered images for novel views by our GeoNeRF model are presented in Figures 3.5 and 3.6. Figure 3.5 includes samples from the real forward-facing LLFF dataset (Mildenhall et al., 2019), and Figure 3.6 contains samples from the NeRF realistic synthetic dataset(Mildenhall et al., 2020). In addition to the rendered images, we also show the rendered depth maps for each novel view. Images indicated by GeoNeRF are rendered by our generalizable model, while images indicated by GeoNeRF$_{10k}$ are rendered after fine-tuning our model on each scene for 10k iterations.

3.7.3 Per-Scene Breakdown

Tables 3.10, 3.11, 3.12, and 3.13 break down the quantitative results presented in Section 3.5 into per-scene metrics. The results are consistent with the aggregate results in Tables 3.2 and 3.3. More specifically, Tables 3.10 and 3.11 include the scenes from the real forward-facing LLFF dataset (Mildenhall et al., 2019), and Tables 3.12 and 3.13 contain the scenes from NeRF realistic synthetic dataset (Mildenhall et al., 2020). As it was already shown in Section 3.5, our generalizable GeoNeRF model outperforms all existing generalizable methods on average, and after fine-tuning, it is on par with per-scene optimized vanilla NeRF (Mildenhall et al., 2020).

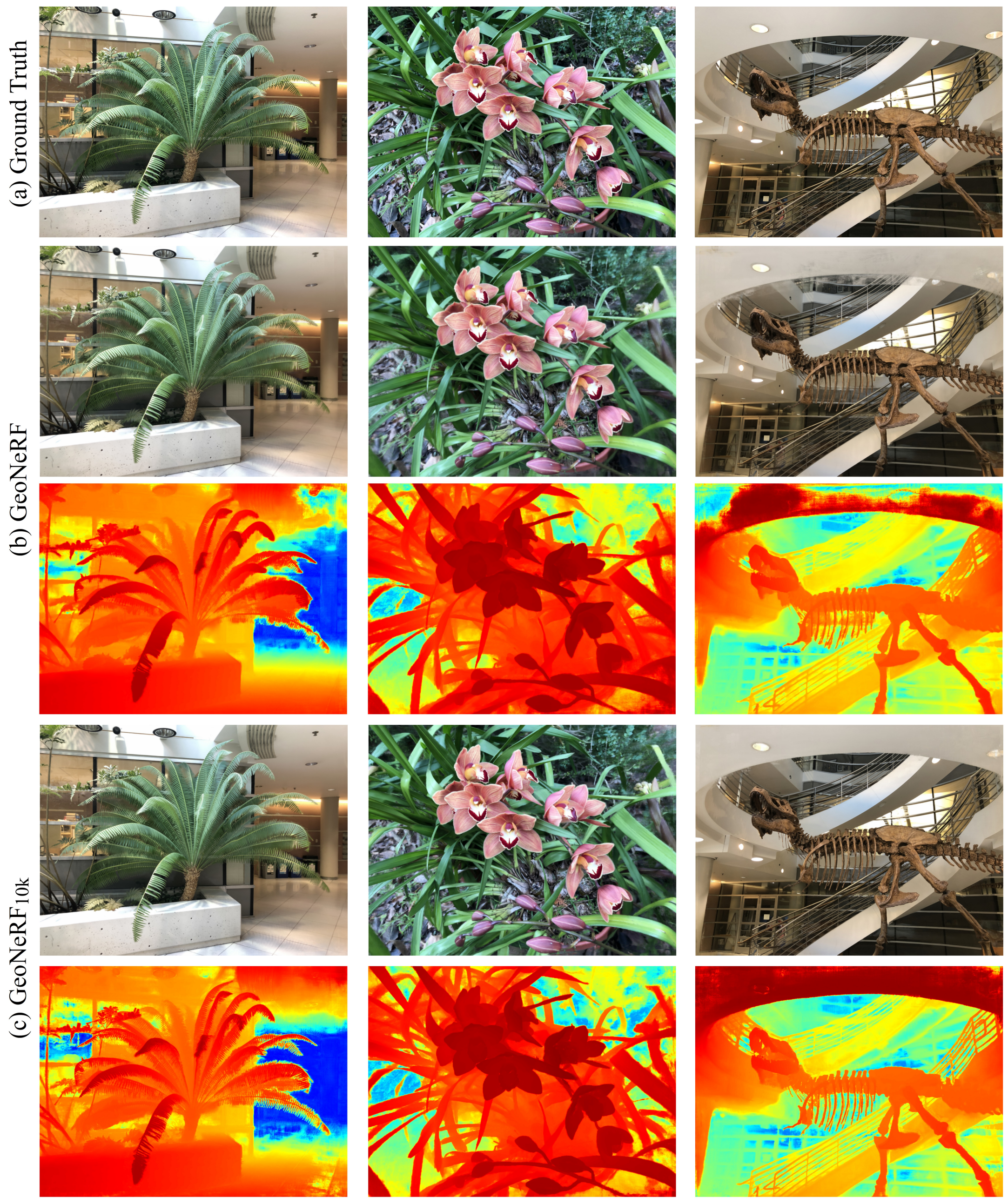


Figure 3.5: Full-size examples of novel images and their depth map rendered by our generalizable (GeoNeRF) and fine-tuned (GeoNeRF$_{10k}$) models. The images are from test scenes of the real forward-facing LLFF dataset (Mildenhall et al., 2019).

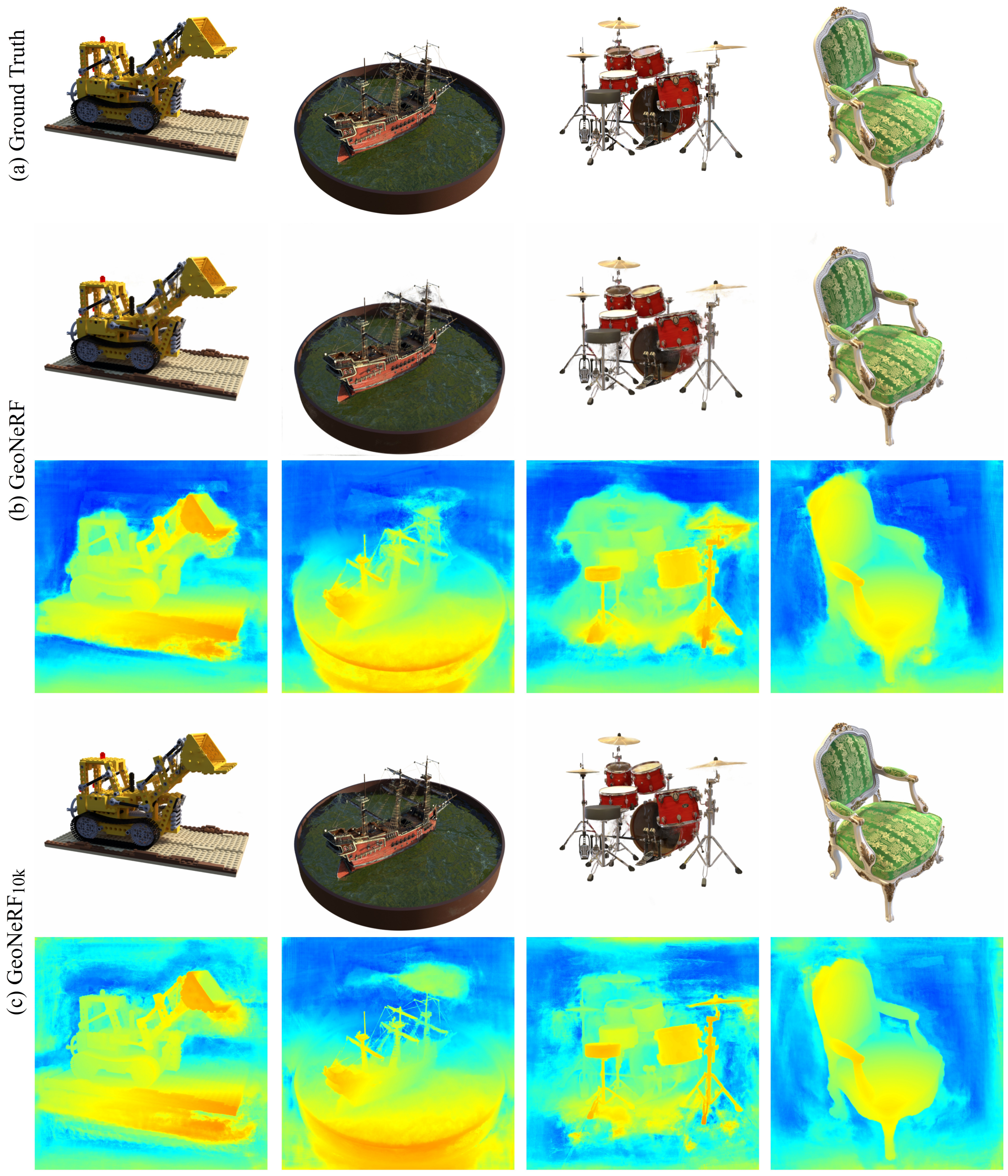


Figure 3.6: Full-size examples of novel images and their depth map rendered by our generalizable (GeoNeRF) and fine-tuned (GeoNeRF$_{10k}$) models. The images are from test scenes of the NeRF realistic synthetic dataset (Mildenhall et al., 2020).

	PSNR↑							
	Fern	Flower	Fortress	Horns	Leaves	Orchids	Room	T-Rex
pixelNeRF	12.40	10.00	14.07	11.07	9.85	9.62	11.75	10.55
IBRNet	23.84	26.67	30.00	26.48	20.19	19.34	**29.94**	**24.57**
MVSNeRF	21.15	24.74	26.03	23.57	17.51	17.85	26.95	23.20
GeoNeRF	**24.61**	**28.12**	**30.49**	**26.96**	**20.58**	**20.24**	28.74	23.75

	SSIM↑							
	Fern	Flower	Fortress	Horns	Leaves	Orchids	Room	T-Rex
pixelNeRF	0.531	0.433	0.674	0.516	0.268	0.317	0.691	0.458
IBRNet	0.772	0.856	0.883	0.869	0.719	0.633	0.946	0.861
MVSNeRF	0.638	**0.888**	0.872	0.868	0.667	0.657	**0.951**	0.868
GeoNeRF	**0.811**	0.885	**0.898**	**0.901**	**0.741**	**0.666**	0.935	**0.877**

	LPIPS↓							
	Fern	Flower	Fortress	Horns	Leaves	Orchids	Room	T-Rex
pixelNeRF	0.650	0.708	0.608	0.705	0.695	0.721	0.611	0.667
IBRNet	0.246	0.164	0.153	0.177	0.230	0.287	0.153	0.230
MVSNeRF	0.238	0.196	0.208	0.237	0.313	0.274	0.172	**0.184**
GeoNeRF	**0.202**	**0.133**	**0.123**	**0.140**	**0.222**	**0.256**	**0.150**	0.212

Table 3.10: Per-scene Quantitative comparison of our proposed GeoNeRF with existing generalizable NeRF models, pixelNeRF (Yu et al., 2021), IBRNet (Wang et al., 2021b), and MVSNeRF (Chen et al., 2021), on real forward-facing LLFF dataset (Mildenhall et al., 2019) in terms of PSNR (higher is better), SSIM (Wang et al., 2004) (higher is better), and LPIPS (Zhang et al., 2018a) (lower is better) metrics.

	PSNR↑							
	Fern	Flower	Fortress	Horns	Leaves	Orchids	Room	T-Rex
NeRF	25.17	27.40	**31.16**	27.45	20.92	20.36	**32.70**	**26.80**
GeoNeRF$_{10k}$	**25.24**	28.57	30.75	**28.12**	**21.40**	20.39	31.51	26.63
GeoNeRF$_{1k}$	25.08	**28.74**	30.83	27.66	21.16	**20.41**	30.52	26.07

	SSIM↑							
	Fern	Flower	Fortress	Horns	Leaves	Orchids	Room	T-Rex
NeRF	0.792	0.827	0.881	0.828	0.690	0.641	0.948	0.880
GeoNeRF$_{10k}$	**0.829**	0.890	0.900	**0.912**	**0.781**	**0.674**	**0.956**	**0.910**
GeoNeRF$_{1k}$	0.824	**0.892**	**0.905**	0.908	0.769	0.673	0.946	0.901

	LPIPS↓							
	Fern	Flower	Fortress	Horns	Leaves	Orchids	Room	T-Rex
NeRF	0.280	0.219	0.171	0.268	0.316	0.321	0.178	0.249
GeoNeRF$_{10k}$	**0.185**	0.120	0.125	**0.126**	**0.183**	**0.247**	**0.126**	**0.181**
GeoNeRF$_{1k}$	0.189	**0.114**	**0.117**	0.130	0.198	0.248	0.135	0.188

Table 3.11: Per-scene Quantitative comparison of our fine-tuned GeoNeRF with per-scene optimized vanilla NeRF (Mildenhall et al., 2020) on real forward-facing LLFF dataset (Mildenhall et al., 2019) in terms of PSNR (higher is better), SSIM (Wang et al., 2004) (higher is better), and LPIPS (Zhang et al., 2018a) (lower is better) metrics. Our model is fine-tuned on each scene for 10k iterations (GeoNeRF$_{10k}$) and 1k iterations (GeoNeRF$_{1k}$), and NeRF is optimized for 200k iterations.

	PSNR↑							
	Chair	Drums	Ficus	Hotdog	Lego	Materials	Mic	Ship
pixelNeRF	7.18	8.15	6.61	6.80	7.74	7.61	7.71	7.30
IBRNet	28.54	21.22	24.23	31.72	24.59	22.20	27.97	23.64
MVSNeRF	23.35	20.71	21.98	28.44	23.18	20.05	22.62	23.35
GeoNeRF	**31.84**	**24.00**	**25.28**	**34.33**	**28.80**	**26.16**	**31.15**	**25.08**

	SSIM↑							
	Chair	Drums	Ficus	Hotdog	Lego	Materials	Mic	Ship
pixelNeRF	0.624	0.670	0.669	0.669	0.671	0.644	0.729	0.584
IBRNet	0.948	0.896	0.915	0.952	0.918	0.905	0.962	0.834
MVSNeRF	0.876	0.886	0.898	0.962	0.902	0.893	0.923	**0.886**
GeoNeRF	**0.973**	**0.921**	**0.931**	**0.975**	**0.956**	**0.926**	**0.978**	0.844

	LPIPS↓							
	Chair	Drums	Ficus	Hotdog	Lego	Materials	Mic	Ship
pixelNeRF	0.386	0.421	0.335	0.433	0.427	0.432	0.329	0.526
IBRNet	0.066	**0.091**	0.097	0.067	0.095	**0.115**	0.051	0.219
MVSNeRF	0.282	0.187	0.211	0.173	0.204	0.216	0.177	0.244
GeoNeRF	**0.040**	0.098	**0.092**	**0.056**	**0.059**	0.116	**0.037**	**0.200**

Table 3.12: Per-scene Quantitative comparison of our proposed GeoNeRF with existing generalizable NeRF models, pixelNeRF (Yu et al., 2021), IBRNet (Wang et al., 2021b), and MVSNeRF (Chen et al., 2021), on NeRF realistic synthetic dataset (Mildenhall et al., 2020) in terms of PSNR (higher is better), SSIM (Wang et al., 2004) (higher is better), and LPIPS (Zhang et al., 2018a) (lower is better) metrics.

	PSNR↑							
	Chair	Drums	Ficus	Hotdog	Lego	Materials	Mic	Ship
NeRF	33.00	25.01	**30.13**	36.18	**32.54**	**29.62**	32.91	28.65
GeoNeRF$_{10k}$	**33.54**	**25.13**	27.79	**36.26**	30.32	28.19	**33.41**	**28.76**
GeoNeRF$_{1k}$	32.76	24.74	27.06	35.71	29.79	27.69	32.83	28.11

	SSIM↑							
	Chair	Drums	Ficus	Hotdog	Lego	Materials	Mic	Ship
NeRF	0.967	0.925	**0.964**	0.974	0.961	0.949	0.980	0.856
GeoNeRF$_{10k}$	**0.980**	**0.935**	0.955	**0.983**	**0.965**	**0.953**	**0.987**	**0.890**
GeoNeRF$_{1k}$	0.977	0.930	0.948	0.982	0.961	0.948	0.985	0.883

	LPIPS↓							
	Chair	Drums	Ficus	Hotdog	Lego	Materials	Mic	Ship
NeRF	0.046	0.091	**0.044**	0.121	0.050	0.063	0.028	0.206
GeoNeRF$_{10k}$	**0.024**	**0.073**	0.061	**0.032**	**0.041**	**0.058**	**0.016**	**0.137**
GeoNeRF$_{1k}$	0.030	0.081	0.069	0.034	0.046	0.069	0.020	0.145

Table 3.13: Per-scene Quantitative comparison of our fine-tuned GeoNeRF with per-scene optimized vanilla NeRF (Mildenhall et al., 2020) on NeRF realistic synthetic dataset (Mildenhall et al., 2020) in terms of PSNR (higher is better), SSIM (Wang et al., 2004) (higher is better), and LPIPS (Zhang et al., 2018a) (lower is better) metrics. Our model is fine-tuned on each scene for 10k iterations (GeoNeRF$_{10k}$) and 1k iterations (GeoNeRF$_{1k}$), and NeRF is optimized for 500k iterations.

3.7.4 Ablation Study

An ablation study of our generalizable model on the NeRF synthetic dataset (Mildenhall et al., 2020) and the real forward-facing dataset (Mildenhall et al., 2019) is presented in Tables 3.14 and 3.15, contrasting the effectiveness of individual components of our proposed model. We evaluated GeoNeRF in the cases where (a) no self-supervision loss is used, (b) no positional encoding is employed, (c) points on a ray are merely sampled uniformly, (d) occluded views are not excluded, (e) attention mechanism is removed from the renderer, (f) view-independent tokens are not regularized with the AE network before predicting volume densities, and (g) only a single cost volume is constructed per-view instead of cascaded multi-level cost volumes.

Figure 3.7 contains examples from the NeRF synthetic dataset (Mildenhall et al., 2020) for qualitative analysis corresponding to the experiments in Table 3.14. The examples focus on challenging views of the scenes in order to contrast the behavior of the models properly.

NeRF Realistic Synthetic Dataset (Mildenhall et al., 2020)				
Experiment	PSNR↑	SSIM↑	LPIPS↓	Examples
a. Without self-supervision	28.10	0.935	0.098	Figure 3.7.a
b. Without positional encoding	27.19	0.927	0.116	Figure 3.7.b
c. Uniform sampling along a ray	28.04	0.934	0.089	Figure 3.7.c
d. Without occlusion masks	27.92	0.932	0.097	Figure 3.7.d
e. Without attention mechanism	27.69	0.929	0.135	Figure 3.7.e
f. Without the AE network	23.53	0.884	0.182	Figure 3.7.f
g. Single cost volume	26.60	0.915	0.132	Figure 3.7.g
h. Full GeoNeRF	**28.33**	**0.938**	**0.087**	Figure 3.7.h

Table 3.14: Ablation study of the key components of GeoNeRF. The evaluation is performed on the NeRF synthetic (Mildenhall et al., 2020) test scenes. See Section 3.7.4 for the details of these experiments, and see Figure 3.7 for qualitative analysis.

Real Forward Facing LLFF Dataset (Mildenhall et al., 2019)			
Experiment	PSNR↑	SSIM↑	LPIPS↓
a. Without self-supervision	25.37	0.836	0.184
b. Without positional encoding	25.02	0.836	0.189
c. Uniform sampling along a ray	25.31	0.835	0.184
d. Without occlusion masks	25.22	0.834	0.185
e. Without attention mechanism	24.95	0.828	0.194
f. Without the AE network	24.92	0.821	0.199
g. Single cost volume	24.60	0.814	0.211
h. Full GeoNeRF	**25.44**	**0.839**	**0.180**

Table 3.15: Ablation study of the key components of GeoNeRF. The same evaluation of the experiments in Table 3.14 is performed on the real forward-facing LLFF (Mildenhall et al., 2019) test scenes. See Section 3.7.4 for the details of these experiments.

3.7.5 Limitations

Our model with the experimental settings in Section 3.5 can be trained and evaluated on a single GPU with 16 GB of memory. Failure cases in our model could occur when the stereo reconstruction fails in the geometry reasoner, and the renderer is misled by incorrect geometry priors. Since the architecture of the geometry reasoner is inspired by multi-view stereo models, it is prone to failure in textureless areas similarly. Such failure examples are shown in Figure 3.8.

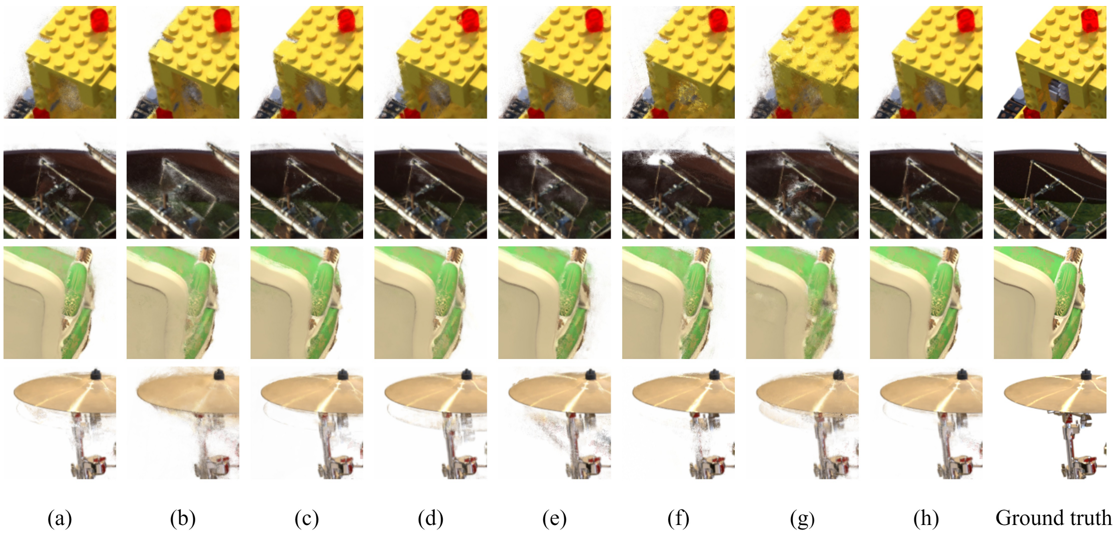


Figure 3.7: Qualitative ablation study of the key components of GeoNeRF. The examples are selected from challenging views of the NeRF synthetic dataset (Mildenhall et al., 2020). Columns correspond to the experiments in Table 3.14.

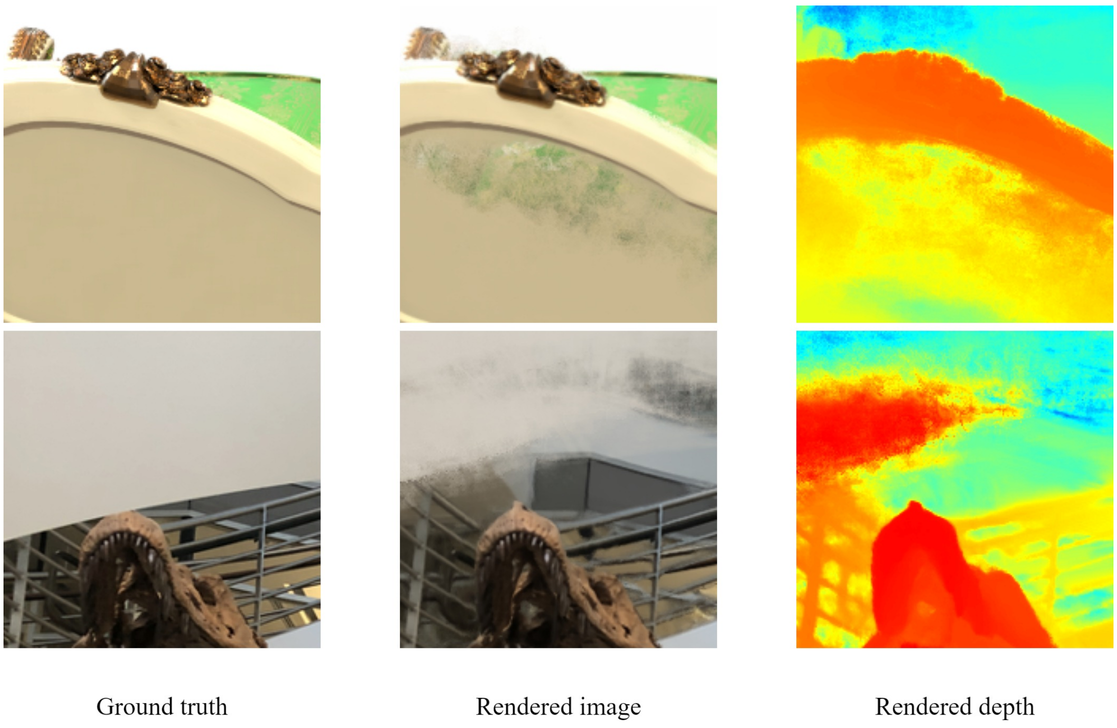


Figure 3.8: Failure examples in our method where stereo reconstruction fails in the geometry reasoner for textureless areas.

4 ESLAM: an Efficient Dense Visual SLAM

Disclaimer: This chapter is adapted from the following article – with permission of all co-authors and the conference:

Johari, M. M., Carta, C., and Fleuret, F. (2023). ESLAM: Efficient Dense SLAM System Based on Hybrid Representation of Signed Distance Fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17408-17419. Selected as a **Highlight Paper** (Top 2.5%).

4.1 Abstract

We present ESLAM, an efficient implicit neural representation method for Simultaneous Localization and Mapping (SLAM). ESLAM reads RGB-D frames with unknown camera poses in a sequential manner and incrementally reconstructs the scene representation while estimating the current camera position in the scene. We incorporate the latest advances in Neural Radiance Fields (NeRF) into a SLAM system, resulting in an efficient and accurate dense visual SLAM method. Our scene representation consists of multi-scale axis-aligned perpendicular feature planes and shallow decoders that, for each point in the continuous space, decode the interpolated features into Truncated Signed Distance Field (TSDF) and RGB values. Our extensive experiments on three standard datasets, Replica, ScanNet, and TUM RGB-D show that ESLAM improves the accuracy of 3D reconstruction and camera localization of state-of-the-art dense visual SLAM methods by more than 50%, while it runs up to $\times 10$ faster and does not require any pre-training. The implementation source code and visualization videos showcasing the obtained results can be accessed through the following link: https://www.idiap.ch/paper/eslam

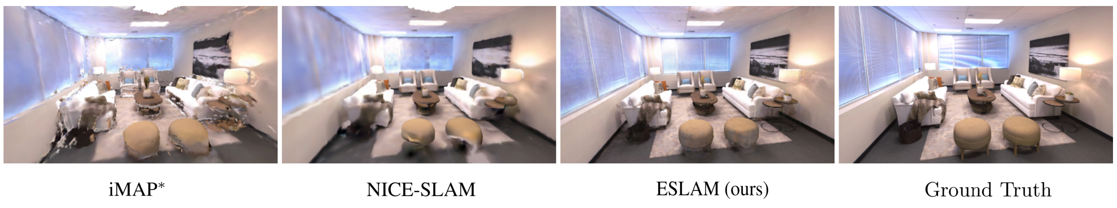


Figure 4.1: Our pre-train-free ESLAM model reconstructs scene details more accurately than existing works: iMAP* (Sucar et al., 2021) and NICE-SLAM (Zhu et al., 2022), while it runs up to ×10 faster (see Section 4.5.2 for runtime analysis). The ground truth image is rendered with ReplicaViewer (Straub et al., 2019).

4.2 Introduction

Dense visual Simultaneous Localization and Mapping (SLAM) is a fundamental challenge in 3D computer vision with several applications such as autonomous driving, robotics, and virtual/augmented reality. It is defined as constructing a 3D map of an unknown environment while simultaneously approximating the camera pose.

While traditional SLAM systems (Mur-Artal and Tardós, 2017a; Engel et al., 2014; Newcombe et al., 2011b; Schops et al., 2019; Whelan et al., 2015, 2012) mostly focus on localization accuracy, recent learning-based dense visual SLAM methods (Bloesch et al., 2018; Yang et al., 2022b; Czarnowski et al., 2020; Sucar et al., 2020; Zhi et al., 2019; Teed and Deng, 2021; McCormac et al., 2017; Sünderhauf et al., 2017; Tang and Tan, 2018; Koestler et al., 2022) provide meaningful global 3D maps and show reasonable but limited reconstruction accuracy.

Following the advent of Neural Radiance Fields (NeRF) (Mildenhall et al., 2020) and the demonstration of their capacity to reason about the geometry of a large-scale scene (Deng et al., 2022; Kosiorek et al., 2021; Chen et al., 2021; Wei et al., 2021; Jain et al., 2021; Wu et al., 2022) and reconstruct 3D surfaces (Yariv et al., 2021; Azinović et al., 2022; Wang et al., 2021a; Sun et al., 2022b; Or-El et al., 2022; Li et al., 2022a; Ortiz et al., 2022; Zhang et al., 2021; Wang et al., 2022b), novel NeRF-based dense SLAM methods have been developed. In particular, iMAP (Sucar et al., 2021) and NICE-SLAM (Zhu et al., 2022) utilize neural implicit networks to achieve a consistent geometry representation.

IMAP (Sucar et al., 2021) represents the geometry with a single huge MLP, similar to vanilla NeRF (Mildenhall et al., 2020), and optimizes the camera poses during the rendering process. NICE-SLAM (Zhu et al., 2022) improves iMAP by storing the representation locally on voxel grids to prevent the forgetting problem. Despite promising reconstruction quality, these methods are computationally demanding for real-time applications, and their ability to capture geometry details is limited. In addition, NICE-SLAM (Zhu et al., 2022) uses frozen pre-trained MLPs, which limits its generalizability to novel scenes. We take NICE-SLAM (Zhu et al., 2022) as a baseline and provide the following contributions:

- We leverage implicit Truncated Signed Distance Field (TSDF) (Azinović et al., 2022) to represent geometry, which converges noticeably faster than the common rendering-based representations like volume density (Sucar et al., 2021) or occupancy (Zhu et al., 2022) and results in higher quality reconstruction.

- Instead of storing features on voxel grids, we propose employing multi-scale axis-aligned feature planes (Chan et al., 2022) which leads to reducing the memory footprint growth rate w.r.t. scene side length from cubic to quadratic.

- We benchmark our method on three challenging datasets, Replica (Straub et al., 2019), ScanNet (Dai et al., 2017), and TUM RGB-D (Sturm et al., 2012), to demonstrate the performance of our method in comparison to existing ones and provide an extensive ablation study to validate our design choices.

Thanks to the inherent smoothness of representing the scene with feature planes, our method produces higher-quality smooth surfaces without employing explicit smoothness loss functions like the one used by Wang et al. (2022a).

Concurrent with our work, the following studies also propose Radiance Fields-based SLAM systems: iDF-SLAM (Ming et al., 2022) also uses TSDF, but it is substantially slower and less accurate than NICE-SLAM (Zhu et al., 2022). Orbeez-SLAM (Chung et al., 2023) operates in real-time at the cost of poor 3D reconstruction. Compromising accuracy and quality, MeSLAM (Kruzhkov et al., 2022) introduces a memory-efficient SLAM. MonoNeuralFusion (Zou et al., 2022) proposes an incremental 3D reconstruction model, assuming that ground truth camera poses are available. Lastly, NeRF-SLAM (Rosinol et al., 2023) presents a monocular SLAM system with hierarchical volumetric Neural Radiance Fields optimized using an uncertainty-based depth loss.

4.3 Related Work

Dense Visual SLAM. The ubiquity of cameras has made visual SLAM a field of major interest in the last decades. Traditional visual SLAM employs pixel-wise optimization of geometric and/or photometric constraints from image information. Depending on the information source, visual SLAM divides into three main categories: visual-only (Newcombe et al., 2011b; Engel et al., 2014; Forster et al., 2014; Mur-Artal and Tardós, 2017a; Tateno et al., 2017; Engel et al., 2017), visual-inertial (Mourikis and Roumeliotis, 2007; Leutenegger et al., 2015; Bloesch et al., 2015; Mur-Artal and Tardós, 2017b; Qin et al., 2018; Von Stumberg et al., 2018) and RGB-D (Newcombe et al., 2011a; Kerl et al., 2013; Endres et al., 2013; Campos et al., 2021) SLAM. Visual-only SLAM uses single or multi-camera setups but presents higher technical challenges compared to others. Visual-inertial information can improve accuracy, but complexifies the system and requires an extra calibration step. The advent of the Kinect brought popularity to RGB-D setups with improved performance but had drawbacks such as larger memory and power requirements, and limitations to indoor settings. Recently, learning-based approaches (Bloesch et al., 2018; Li et al., 2018a, 2020; Czarnowski et al., 2020; Teed and Deng, 2021) have made great advances in the field, improving both accuracy and robustness compared to traditional methods.

Neural Implicit 3D Reconstruction. Neural Radiance Fields (NeRF) have impacted 3D Computer Vision applications, such as novel view synthesis (Mildenhall et al., 2020; Martin-Brualla et al., 2021; Verbin et al., 2022; Mildenhall et al., 2022), surface reconstruction (Park et al., 2019; Yariv et al., 2021; Oechsle et al., 2021; Wang et al., 2021a; Zhang et al., 2021; Wang et al., 2022b; Yariv et al., 2020), dynamic scene representation (Gao et al., 2021; Park et al., 2021a; Pumarola et al., 2021; Park et al., 2021b), and camera pose estimation (Yen-Chen et al., 2021; Wang et al., 2021c; Lin et al., 2021; Jeong et al., 2021; Xia et al., 2022). The exploitation of neural implicit representations for 3D reconstruction at real-world scale is studied in many recent studies (Azinović et al., 2022; Wang et al., 2022a; Bozic et al., 2021; Choe et al., 2021; Murez et al., 2020; Sun et al., 2021; Weder et al., 2021; Yan et al., 2021; Li et al., 2022a). The most related works to ours are iMAP (Sucar et al., 2021) and NICE-SLAM (Zhu et al., 2022). IMAP presents a NeRF-style dense SLAM system. NICE-SLAM extends iMAP by modeling the scene with voxel grid features and decoding them into occupancies using pre-trained MLPs. However, the generalizability of NICE-SLAM to novel scenes is limited because of the frozen pre-trained MLPs. Another issue is the cubic memory growth rate of their model, which results in using low-resolution voxel grids and losing fine geometry details. In contrast, we employ compact plane-based features (Chan et al., 2022) which are directly decoded to TSDF, improving both efficiency and accuracy of localization and reconstruction.

4.4 Method

The overview of our method is shown in Figure 4.2. Given a set of sequential RGB-D frames $\{I_i, D_i\}_{i=1}^{M}$, our model predicts camera poses $\{R_i|t_i\}_{i=1}^{M}$ and an implicit TSDF $\boldsymbol{\phi_g}$ representation that can be used in the marching cubes algorithm (Lorensen and Cline, 1987) to extract 3D meshes. We expect TSDF to denote the distance to the closest surface with a positive sign in the free space and a negative sign inside the surfaces. We employ normalized TSDF, such that it is zero on the surfaces and has a magnitude of one at the truncation distance T, which is a hyper-parameter. Section 4.4.1 describes how we represent a scene with axis-aligned feature planes. Section 4.4.2 walks through the rendering process, which converts raw representations into pixel depths and colors. Section 4.4.3 introduces our loss functions. Finally, Section 4.4.4 provides the details of localization and reconstruction in our SLAM system.

4.4.1 Axis-Aligned Feature Planes

Although voxel grid-based NeRF architectures (Fridovich-Keil et al., 2022; Sun et al., 2022a; Wang et al., 2022a) exhibit rapid convergence, they struggle with cubical memory growing.

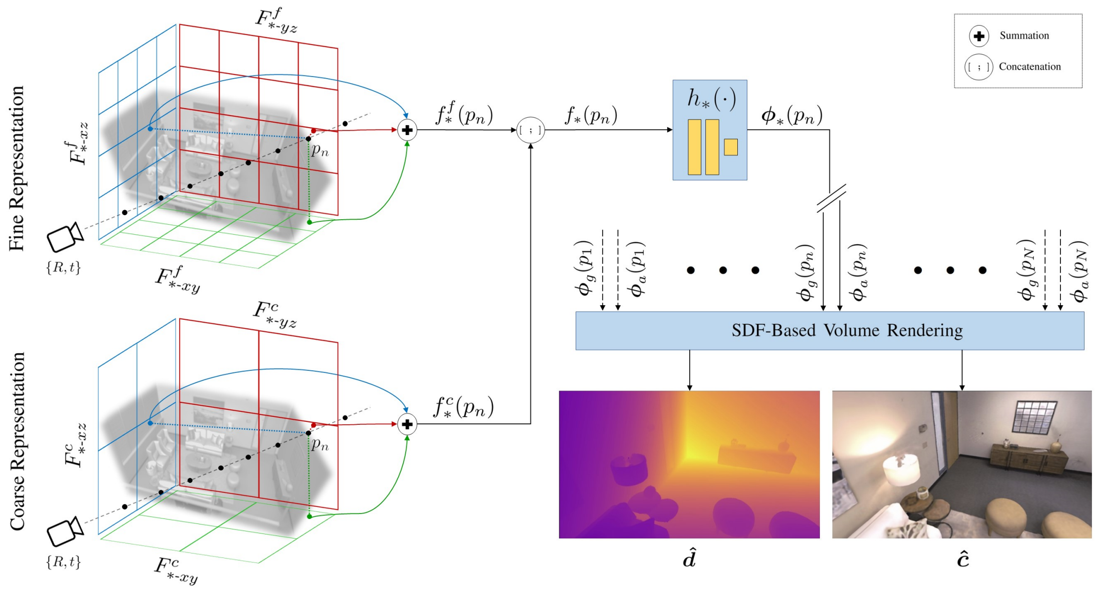


Figure 4.2: The overview of ESLAM. Given the symmetry of our processes for geometry and appearance, we exhibit both processes on the same pipeline for simplicity. The symbol $*$ represents both geometry g and appearance a, *e.g.*, $f_*(p_n)$ can be either $f_g(p_n)$ or $f_a(p_n)$. At an estimated camera pose $\{R,t\}$, we cast a ray for each pixel and sample N points $\{p_n\}_{n=1}^N$ along it (Section 4.4.2). Each point p_n is projected onto the coarse and fine feature planes and bilinearly interpolates the four nearest neighbor features on each plane (Section 4.4.1). The interpolated features at each level are added together, and the results from both levels are concatenated together to form the inputs $\{\boldsymbol{f_g}(p_n), \boldsymbol{f_a}(p_n)\}$ of the decoders $\{h_g, h_a\}$ (Section 4.4.1). The geometry decoder h_g estimates TSDF $\boldsymbol{\phi_g}(p_n)$ based on $\boldsymbol{f_g}(p_n)$, and the appearance decoder h_a estimates the raw color $\boldsymbol{\phi_a}(p_n)$ based on $\boldsymbol{f_a}(p_n)$ for each point p_n (Section 4.4.1). Once TSDFs and raw colors of all points on a ray are generated, our SDF-based rendering process estimates the depth $\hat{\boldsymbol{d}}$ and the color $\hat{\boldsymbol{c}}$ for each pixel (Section 4.4.2).

Different solutions have been proposed to mitigate the memory growth issue (Müller et al., 2022; Chen et al., 2022; Chan et al., 2022). Inspired by Chan et al. (2022), we employ a tri-plane architecture (see Figure 4.2), in which we store and optimize features on perpendicular axis-aligned planes. Mimicking the trend in voxel-based methods (Müller et al., 2022; Sun et al., 2022a; Chen et al., 2022), we propose using feature planes at two scales, *i.e.*, coarse and fine. Coarse-level representation allows efficient reconstruction of free space with fewer sample points and optimization iterations. Moreover, we suggest employing separate feature planes for representing geometry and appearance, which mitigates the forgetting problem for geometry reconstruction since appearance fluctuates more frequently in a scene than geometry.

Specifically, we use three coarse feature planes $\{F^c_{g\text{-}xy}, F^c_{g\text{-}xz}, F^c_{g\text{-}yz}\}$ and three fine ones $\{F^f_{g\text{-}xy}, F^f_{g\text{-}xz}, F^f_{g\text{-}yz}\}$ for representing the geometry. Similarly, three coarse $\{F^c_{a\text{-}xy}, F^c_{a\text{-}xz}, F^c_{a\text{-}yz}\}$ and three fine $\{F^f_{a\text{-}xy}, F^f_{a\text{-}xz}, F^f_{a\text{-}yz}\}$ planes are used for representing appearance of a scene. This architecture prevents model size from growing cubically with the scene side-length as is the case for voxel-based models.

To reason about the geometry of a point p in the continuous space, we first project it onto all the geometry planes. The geometry feature $\boldsymbol{f_g}(p)$ for point p is then formed by 1) bilinearly interpolating the four nearest neighbors on each feature plane, 2) summing the interpolated coarse features and the fine ones respectively into the coarse output $f^c_g(p)$ and fine output $f^f_g(p)$, and 3) concatenating the outputs together. Formally:

$$\begin{aligned}
f^c_g(p) &= F^c_{g\text{-}xy}(p) + F^c_{g\text{-}xz}(p) + F^c_{g\text{-}yz}(p) \\
f^f_g(p) &= F^f_{g\text{-}xy}(p) + F^f_{g\text{-}xz}(p) + F^f_{g\text{-}yz}(p) \\
\boldsymbol{f_g}(p) &= [f^c_g(p); f^f_g(p)]
\end{aligned} \tag{4.1}$$

The appearance feature $\boldsymbol{f_a}(p)$ is obtained similarly:

$$\begin{aligned}
f^c_a(p) &= F^c_{a\text{-}xy}(p) + F^c_{a\text{-}xz}(p) + F^c_{a\text{-}yz}(p) \\
f^f_a(p) &= F^f_{a\text{-}xy}(p) + F^f_{a\text{-}xz}(p) + F^f_{a\text{-}yz}(p) \\
\boldsymbol{f_a}(p) &= [f^c_a(p); f^f_a(p)]
\end{aligned} \tag{4.2}$$

These features are decoded into TSDF $\boldsymbol{\phi_g}(p)$ and raw color $\boldsymbol{\phi_a}(p)$ values via shallow two-layer MLPs $\{h_g, h_a\}$:

$$\boldsymbol{\phi_g}(p) = h_g\left(\boldsymbol{f_g}(p)\right) \quad \text{and} \quad \boldsymbol{\phi_a}(p) = h_a\left(\boldsymbol{f_a}(p)\right) \tag{4.3}$$

These raw TSDF and color outputs can be utilized for depth/color rendering as well as mesh extraction.

4.4.2 SDF-Based Volume Rendering

When processing input frame i, emulating the ray casting in NeRF (Mildenhall et al., 2020), we select random pixels and calculate their corresponding rays using the current estimate of the camera pose $\{R_i|t_i\}$. For rendering the depths and colors of the rays, we first sample N_{strat} samples on each ray by stratified sampling and then sample additional N_{imp} points near surfaces. For pixels with ground truth depths, the N_{imp} additional points are sampled uniformly inside the truncation distance T w.r.t. the depth measurement, whereas for other pixels, N_{imp} points are sampled with the importance sampling technique (Mildenhall et al., 2020; Martin-Brualla et al., 2021; Wang et al., 2022a; Sucar et al., 2021) based on the weights computed for the stratified samples.

For all $N = N_{strat} + N_{imp}$ points on a ray $\{p_n\}_{n=1}^{N}$, we query TSDF $\boldsymbol{\phi_g}(p_n)$ and raw color $\boldsymbol{\phi_a}(p_n)$ from our networks and use the SDF-Based rendering approach in StyleSDF (Or-El et al., 2022) to convert SDF values to volume densities:

$$\boldsymbol{\sigma}(p_n) = \beta \cdot \text{Sigmoid}\left(-\beta \cdot \boldsymbol{\phi_g}(p_n)\right) \tag{4.4}$$

where β is a learnable parameter that controls the sharpness of the surface boundary. Negative values of SDF push Sigmoid toward one, resulting in volume density inside the surface. The volume density then is used for rendering the color and depth of each ray:

$$\begin{gathered} w_n = \exp\left(-\sum_{k=1}^{n-1} \boldsymbol{\sigma}(p_k)\right)\left(1 - \exp\left(-\boldsymbol{\sigma}\left(p_n\right)\right)\right) \\ \hat{\boldsymbol{c}} = \sum_{n=1}^{N} w_n \boldsymbol{\phi_a}(p_n) \quad \text{and} \quad \hat{\boldsymbol{d}} = \sum_{n=1}^{N} w_n z_n \end{gathered} \tag{4.5}$$

where z_n is the depth of point p_n w.r.t.the camera pose.

4.4.3 Loss Functions

One advantage of TSDF over other representations, such as occupancy, is that it allows us to use per-point losses, along with rendering ones. These losses account for the rapid convergence of our model. Following the practice of Azinović et al. (2022), assuming a batch of rays R with ground truth depths are selected, we define the free space loss as:

$$\mathcal{L}_{fs} = \frac{1}{|R|} \sum_{r \in R} \frac{1}{|P_r^{fs}|} \sum_{p \in P_r^{fs}} (\boldsymbol{\phi_g}(p) - 1)^2 \tag{4.6}$$

where P_r^{fs} is a set of points on the ray r that lie between the camera center and the truncation region of the surface measured by the depth sensor. This loss function encourages TSDF $\boldsymbol{\phi_g}$ to have a value of one in the free space.

For sample points close to the surface and within the truncation region, we use the signed distance objective, which leverages the depth sensor measurement to approximate the signed distance field:

$$\mathcal{L}_T(P_r^T) = \frac{1}{|R|} \sum_{r \in R} \frac{1}{|P_r^T|} \sum_{p \in P_r^T} \left(z(p) + \boldsymbol{\phi_g}(p) \cdot T - D(r)\right)^2 \tag{4.7}$$

where $z(p)$ is the planar depth of point p w.r.t. camera, T is the truncation distance, $D(r)$ is the ray depth measured by the sensor, and P_r^T is a set of points on the ray r that lie in the truncation region, *i.e.*, $|z(p) - D(r)| < T$. We apply the same loss to all points in the truncation region, but we differentiate the importance of points that are closer to the surface in the middle of the truncation region $P_r^{T\text{-}m}$ from those that are at the tail of the truncation region $P_r^{T\text{-}t}$. Formally, we define $P_r^{T\text{-}m}$ as a set of points that $|z(p) - D(r)| < 0.4T$, and define $P_r^{T\text{-}t} = P_r^T - P_r^{T\text{-}m}$, then:

$$\mathcal{L}_{T\text{-}m} = \mathcal{L}_T(P_r^{T\text{-}m}) \quad \text{and} \quad \mathcal{L}_{T\text{-}t} = \mathcal{L}_T(P_r^{T\text{-}t}) \tag{4.8}$$

This enables us to decrease the importance of $\mathcal{L}_{T\text{-}t}$ in mapping, which leads to having a smaller effective truncation distance, reducing artifacts in occluded areas, and reconstructing with higher accuracy while leveraging the entire truncation distance in camera tracking.

In addition to these two per-point loss functions, we also employ reconstruction losses. For pixels with ground truth depths, we impose consistency between the rendered depth and the depth measured by the sensor:

$$\mathcal{L}_d = \frac{1}{|R|} \sum_{r \in R} \left(\hat{\boldsymbol{d}}(r) - D(r) \right)^2 \tag{4.9}$$

Similarly, we impose consistency between the pixel colors and rendered colors:

$$\mathcal{L}_c = \frac{1}{|R|} \sum_{r \in R} (\hat{\boldsymbol{c}}(r) - I(r))^2 \tag{4.10}$$

where $I(r)$ is the pixel color of ray r.

The global loss function of our method is defined as:

$$\mathcal{L} = \lambda_{fs}\mathcal{L}_{fs} + \lambda_{T\text{-}m}\mathcal{L}_{T\text{-}m} + \lambda_{T\text{-}t}\mathcal{L}_{T\text{-}t} + \lambda_d\mathcal{L}_d + \lambda_c\mathcal{L}_c \tag{4.11}$$

where $\{\lambda_{fs}, \lambda_{T\text{-}m}, \lambda_{T\text{-}t}, \lambda_d, \lambda_c\}$ are the weighting coefficients. Note that $\mathcal{L}_c$ is defined on all rays in a training batch, while other losses are only imposed on rays with ground truth measured depths. The global objective is the same for both mapping and tracking in our method, but the weighting coefficients are different.

4.4.4 Mapping and Tracking

Mapping. Our scene representation, *i.e.*, the feature planes and MLP decoders, are randomly initialized at the beginning. With the first input frame $\{I_0, D_0\}$, we fix the camera pose and optimize the feature planes and MLP decoders to best represent the first frame. For subsequent inputs, we update the scene representation iteratively every k frames, and add the latest frame to the global keyframe list, following the practice in iMAP (Sucar et al., 2021) and NICE-SLAM (Zhu et al., 2022). For mapping, we first choose $|R|$ pixels randomly from W frames, which include the current frame, the previous two keyframes, and $W - 3$ frames randomly selected from the keyframe list. Then, we jointly optimize the feature planes, MLP decoders, and camera poses of the W selected frames using the loss functions introduced in Section 4.4.3. Unlike NICE-SLAM (Zhu et al., 2022), our method does not require a staged-optimization policy, and we simply optimize all scene parameters and camera poses

Method	Reconstruction (cm)			
	Depth L1↓	Acc.↓	Comp.↓	Comp. Ratio (%)↑
iMAP*	8.23 ± 0.88	7.16 ± 0.26	5.83 ± 0.27	67.17 ± 2.70
NICE-SLAM	3.29 ± 0.33	1.66 ± 0.07	1.63 ± 0.05	96.74 ± 0.36
ESLAM (ours)	**1.18** ± **0.05**	**0.97** ± **0.02**	**1.05** ± **0.01**	**98.60** ± **0.07**
	Localization (cm)			
	ATE Mean↓	ATE RMSE↓		
iMAP*	2.59 ± 0.58	3.42 ± 0.87		
NICE-SLAM	1.56 ± 0.29	2.05 ± 0.45		
ESLAM (ours)	**0.52** ± **0.03**	**0.63** ± **0.05**		

Table 4.1: Quantitative comparison of our proposed ESLAM with existing NeRF-based dense visual SLAM models, iMAP* (Sucar et al., 2021) and NICE-SLAM (Zhu et al., 2022), on the Replica dataset (Straub et al., 2019) for both reconstruction and localization accuracy. The results are the average and standard deviation of five runs on eight scenes of the Replica dataset. Our method outperforms previous works by a high margin and has lower variances, indicating it is also more stable from run to run. The evaluation metrics for reconstruction are L1 loss (cm) between rendered and ground truth depth maps of 1000 random camera poses, as well as reconstruction accuracy (cm), reconstruction completion (cm), and completion ratio (%). The evaluation metrics for localization are mean and RMSE of ATE (cm) (Sturm et al., 2012). For the details of the evaluations for each scene, refer to the supplementary material in Section 4.7.4. It should also be noted that our method runs up to ×10 faster on this dataset (see Section 4.5.2 for runtime analysis).

simultaneously.

Tracking. The localization process of our method is initiated for each input frame. The current estimate of the camera parameters, represented by translation vectors and quaternion rotations (Shoemake, 1985) $\{R|t\}$, are optimized solely based on our global loss function (see Section 4.4.3) with the gradient-based Adam optimizer (Kingma and Ba, 2014). No second-order optimizers or manifold operations are employed for camera tracking in our method. We exclude rays with no ground truth depths and outlier pixels from each optimization step. A pixel is considered an outlier if the difference between its measured depth and rendered depth is ten times greater than the batch's median rendered depth error.

4.5 Experiments

In this section, we validate that our method outperforms existing implicit representation-based methods in both localization and reconstruction accuracy on three standard benchmarks while running up to ×10 faster.

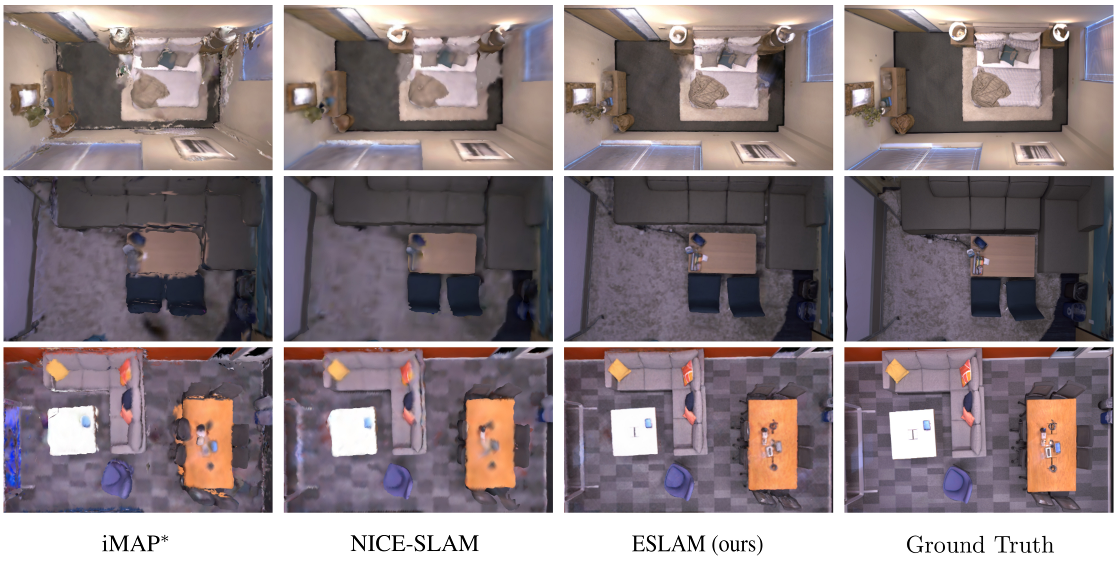


Figure 4.3: Qualitative comparison of our proposed ESLAM method's geometry reconstruction with existing NeRF-based dense visual SLAM models, iMAP* (Sucar et al., 2021) and NICE-SLAM (Zhu et al., 2022), on the Replica dataset (Straub et al., 2019). Our method produces more accurate detailed geometry as well as higher-quality textures. The ground truth images are rendered with the ReplicaViewer software (Straub et al., 2019). It should also be noted that our method runs up to ×10 faster on this dataset (see Section 4.5.2 for runtime analysis). For further qualitative analysis on this dataset, refer to the supplementary material in Section 4.7.3.

Baselines. We compare our method to two existing state-of-the-art NeRF-based dense visual SLAM methods: iMAP (Sucar et al., 2021) and NICE-SLAM (Zhu et al., 2022). Because iMAP is not open source, we use the iMAP* model in our experiment, which is the reimplementation of iMAP by Zhu et al. (2022).

Datasets. We evaluate our method on three standard 3D benchmarks: Replica (Straub et al., 2019), ScanNet (Dai et al., 2017), and TUM RGB-D (Sturm et al., 2012) datasets. We select the same scenes for evaluation as in NICE-SLAM (Zhu et al., 2022).

Metrics. We borrow our evaluation metrics from NICE-SLAM (Zhu et al., 2022). For evaluating scene geometry, we use both 2D and 3D metrics. For the 2D metric, we render depth maps from 1000 random camera poses in each scene and calculate the L1 difference between depths from ground truth meshes and the reconstructed ones. For the 3D metrics, we consider reconstruction accuracy [cm], reconstruction completion [cm], and completion ratio [< 5 cm %]. For evaluating these metrics, we build a TSDF volume for a scene with a resolution of 1 cm and use the marching cubes algorithm (Lorensen and Cline, 1987) to obtain scene meshes. Before evaluating the 3D metrics for our method and for the baselines, we perform mesh culling as recommended by Azinović et al. (2022) and Wang et al. (2022a). For this purpose, we remove faces from a mesh that are not inside any camera frustum or are occluded in all

Scene	ATE	iMAP*	NICE-SLAM	ESLAM (ours)
Sc. 0000	Mean	34.2 ± 12.8	9.9 ± 0.4	**6.5** ± **0.1**
	RMSE	42.7 ± 16.6	12.0 ± 0.5	**7.3** ± **0.2**
Sc. 0059	Mean	13.0 ± 2.4	11.9 ± 1.8	**6.4** ± **0.4**
	RMSE	17.8 ± 7.4	14.0 ± 1.8	**8.5** ± **0.5**
Sc. 0106	Mean	12.9 ± 1.7	7.0 ± 0.2	**6.7** ± **0.1**
	RMSE	15.0 ± 1.7	7.9 ± 0.2	**7.5** ± **0.1**
Sc. 0169	Mean	33.6 ± 15.3	9.2 ± 1.0	**5.9** ± **0.1**
	RMSE	39.1 ± 18.2	10.9 ± 1.1	**6.5** ± **0.1**
Sc. 0181	Mean	20.8 ± 3.8	12.2 ± 0.3	**8.3** ± **0.2**
	RMSE	24.7 ± 5.8	13.4 ± 0.3	**9.0** ± **0.2**
Sc. 0207	Mean	18.6 ± 6.0	5.5 ± 0.3	**5.4** ± **0.1**
	RMSE	20.1 ± 6.8	6.2 ± 0.4	**5.7** ± **0.1**
Average	Mean	22.2 ± 7.0	9.3 ± 0.7	**6.5** ± **0.2**
	RMSE	26.6 ± 9.4	10.7 ± 0.7	**7.4** ± **0.2**

Table 4.2: Quantitative comparison of our proposed ESLAM method's localization accuracy with existing NeRF-based dense visual SLAM models, iMAP* (Sucar et al., 2021) and NICE-SLAM (Zhu et al., 2022), on the ScanNet dataset (Dai et al., 2017). The results are the average and standard deviation of five runs on each scene of ScanNet (Dai et al., 2017). Our method outperforms previous works and has lower variances, indicating it is also more stable from run to run. The evaluation metrics for localization are mean and RMSE of ATE (cm) (Sturm et al., 2012). It should also be noted that our method runs up to ×6 faster on this dataset (see Section 4.5.2 for runtime analysis).

RGB-D frames. For evaluating camera localization, we use ATE (Sturm et al., 2012).

Implementation Details. The truncation distance T is set to 6 cm in our method. We employ coarse feature planes with a resolution of 24 cm for both geometry and appearance. For fine feature planes, we use a resolution of 6 cm for geometry and 3 cm for appearance. All feature planes have 32 channels, resulting in a 64-channel concatenated feature input for the decoders. The decoders are two-layer MLPs with 32 channels in the hidden layer. For the Replica (Straub et al., 2019) dataset, we sample $N_{strat} = 32$ points for stratified sampling and $N_{imp} = 8$ points for importance sampling on each ray. And for the ScanNet (Dai et al., 2017) and TUM RGB-D (Sturm et al., 2012) datasets, we set $N_{strat} = 48$ and $N_{imp} = 8$.

We use different set of loss coefficients for mapping and tracking. During mapping we set $\lambda_{fs} = 5$, $\lambda_{T\text{-}m} = 200$, $\lambda_{T\text{-}t} = 10$, $\lambda_d = 0.1$, and $\lambda_c = 5$. And during tracking, we set $\lambda_{fs} = 10$, $\lambda_{T\text{-}m} = 200$, $\lambda_{T\text{-}t} = 50$, $\lambda_d = 1$, and $\lambda_c = 5$. These coefficients are obtained by performing grid search in our experiments. For further details of our implementation, refer to the supplementary material in Section 4.7.1.

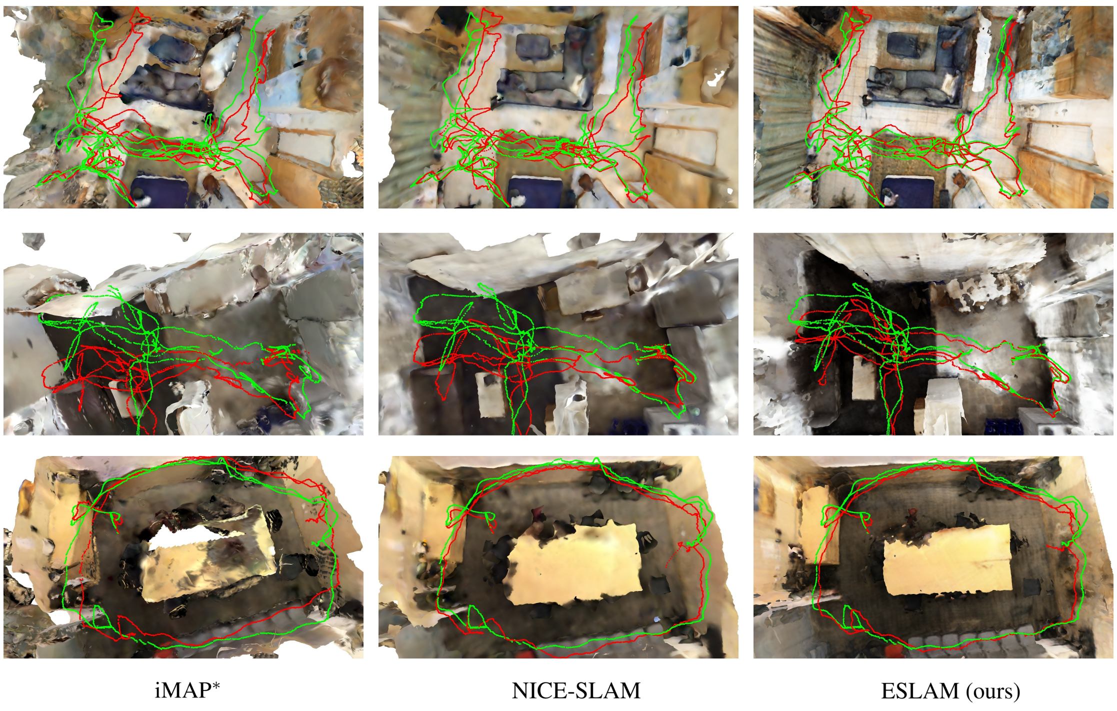


Figure 4.4: Qualitative comparison of our proposed ESLAM method's localization accuracy with existing NeRF-based dense visual SLAM models, iMAP* (Sucar et al., 2021) and NICE-SLAM (Zhu et al., 2022), on the ScanNet dataset (Dai et al., 2017). The ground truth camera trajectory is shown in green, and the estimated trajectory is shown in red. Our method predicts more accurate camera trajectories and does not suffer from drifting issues. It should also be noted that our method runs up to ×6 faster on this dataset (see Section 4.5.2 for runtime analysis).

4.5.1 Experimental Results

Evaluation on Replica (Straub et al., 2019). We provide the quantitative analysis of our experimental results on eight scenes of the Replica dataset (Straub et al., 2019) in Table 4.1. The numbers represent the average and standard deviation of the metrics for five independent runs. As shown in Table 4.1, our method outperforms the baselines for both reconstruction and localization accuracy. Our method also has lower variances, indicating that it is more stable and more robust than existing methods.

Qualitative analysis on the Replica dataset (Straub et al., 2019) is provided in Figure 4.3. The results show that our method reconstructs the details of the scenes more accurately and produces fewer artifacts. Although it is not the focus of this study, our method also produces higher-quality colors for the reconstructed meshes.

Evaluation on ScanNet (Dai et al., 2017). We also benchmark ours and existing methods on multiple large scenes from ScanNet (Dai et al., 2017) to evaluate and compare their scalability.

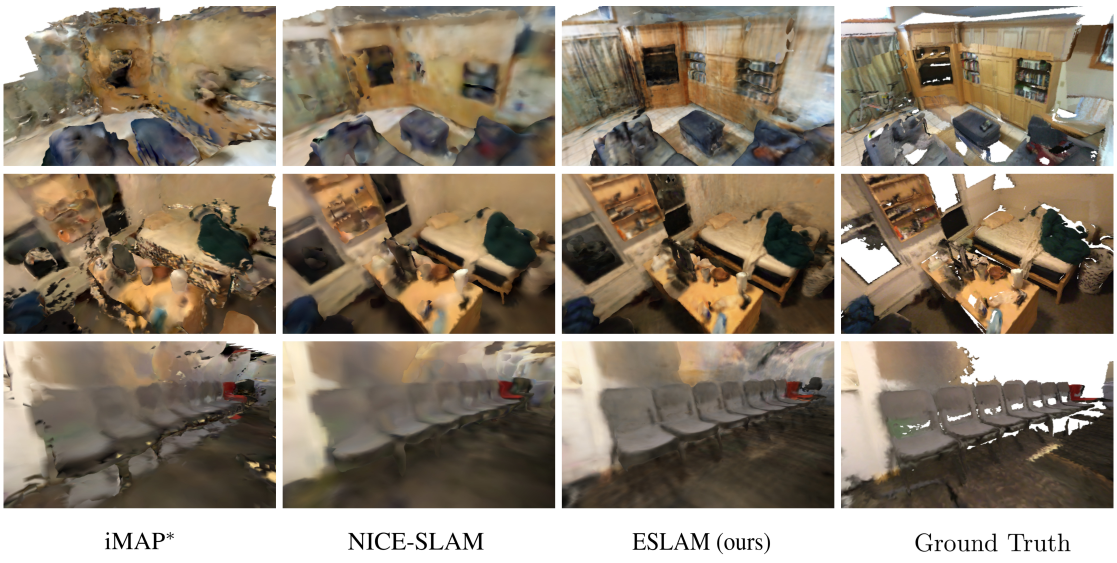


Figure 4.5: Qualitative comparison of our proposed ESLAM method's geometry reconstruction with existing NeRF-based dense visual SLAM models, iMAP* (Sucar et al., 2021) and NICE-SLAM (Zhu et al., 2022), on the ScanNet dataset (Dai et al., 2017). Our method produces more accurate detailed geometry as well as higher-quality textures. The appearance of white backgrounds in ground truth meshes is due to the fact that the ground truth meshes of the ScanNet dataset are incomplete. It should also be noted that our method runs up to ×6 faster on this dataset (see Section 4.5.2 for runtime analysis).

For evaluating camera localization, we conduct five independent experiments on each scene and report the average and standard deviation of the mean and RMSE of ATE (Sturm et al., 2012) in Table 4.2. As demonstrated in the table, our method's localization is more accurate than existing methods. Our method is also considerably more stable from run to run as it has much lower standard deviations. We provide qualitative analysis of camera localization, along with geometry reconstruction, in Figure 4.4. The results show that our method does not suffer from any large drifting and is more robust than existing methods.

Since the ground truth meshes of the ScanNet dataset (Dai et al., 2017) are incomplete, we only provide qualitative analysis for geometry reconstruction, similar to previous works. The qualitative comparison in Figure 4.5 validates that our model reconstructs more precise geometry and detailed textures compared to existing approaches.

Evaluation on TUM RGB-D (Sturm et al., 2012). To further contrast the robustness of our method with the existing ones, we conduct an evaluation study on the real-world TUM RGB-D dataset (Sturm et al., 2012). Since there are no ground truth meshes for the scenes in this dataset, we only present the localization results in Table 4.3 and the qualitative analysis of rendered meshes in Figure 4.6.

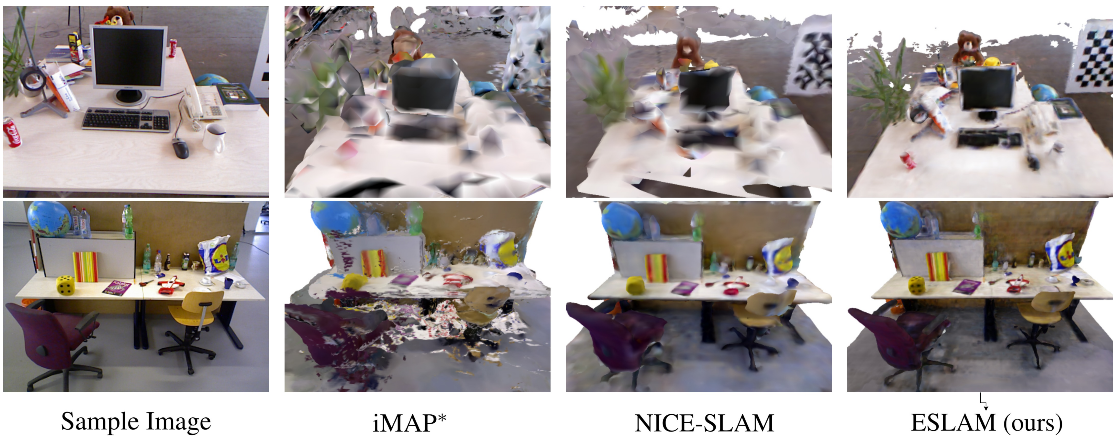


Figure 4.6: Qualitative comparison of our proposed ESLAM method's geometry reconstruction with existing NeRF-based dense visual SLAM models, iMAP* (Sucar et al., 2021) and NICE-SLAM (Zhu et al., 2022), on the TUM RGB-D dataset (Sturm et al., 2012). Our method produces more accurate detailed geometry as well as higher-quality textures. Since there are no ground truth meshes for this dataset, we depict a sample input image.

	fr1/desk	fr2/xyz	fr3/office
iMAP* (Sucar et al., 2021)	4.90	2.05	5.80
NICE-SLAM (Zhu et al., 2022)	2.85	2.39	3.02
ESLAM (ours)	**2.47**	**1.11**	**2.42**

Table 4.3: Quantitative comparison of our proposed ESLAM method's localization accuracy with existing NeRF-based dense visual SLAM models on the TUM RGB-D dataset (Sturm et al., 2012). The evaluation metric is ATE RMSE↓ (cm) (Sturm et al., 2012).

4.5.2 Runtime Analysis

We evaluate the speed and size of our method in comparison with existing approaches in Table 4.4. We report the average frame processing time (FPT), the number of parameters of the model, and memory growth rate w.r.t. scene side length for the scenes `room0` of Replica (Straub et al., 2019) and `scene0000` of ScanNet (Dai et al., 2017) datasets. All methods are benchmarked with an NVIDIA GeForce RTX 3090 GPU. The results indicate that our method is significantly faster than previous works on both datasets. Furthermore, in contrast to NICE-SLAM (Zhu et al., 2022), our model size is smaller and does not grow cubically with the scene side length.

4.6 Conclusion

We presented ESLAM, a dense visual SLAM approach that leverages the latest advances in the Neural Radiance Fields study to improve both the speed and accuracy of neural implicit-based

	Method	Speed FPT (s)	Memory # Parameters	Memory Growth Rate
Replica	iMAP* (Sucar et al., 2021)	5.20	**0.22 M**	?
	NICE-SLAM (Zhu et al., 2022)	2.10	12.18 M	$O(L^3)$
	ESLAM (ours)	**0.18**	6.79 M	$O(L^2)$
ScanNet	iMAP* (Sucar et al., 2021)	5.20	**0.22 M**	?
	NICE-SLAM (Zhu et al., 2022)	3.35	22.04 M	$O(L^3)$
	ESLAM (ours)	**0.55**	17.63 M	$O(L^2)$

Table 4.4: Runtime analysis of our method in comparison with existing ones in terms of average frame processing time (FPT), number of parameters, and model size growth rate w.r.t. scene side length L. All methods are benchmarked with an NVIDIA GeForce RTX 3090 GPU on `room0` of Replica (Straub et al., 2019) and `scene0000` of ScanNet (Dai et al., 2017). Our method is significantly faster and does not grow cubically in size w.r.t. scene side length L. Note that iMAP (Sucar et al., 2021) represents a whole scene in a single MLP, hence its small number of parameters. Accordingly, the scalability and growth rate of iMAP w.r.t. the scene side length L are also unclear.

SLAM systems. We proposed replacing the voxel grid representation with axis-aligned feature planes to prevent the model size from growing cubically with respect to the scene side length. We also demonstrated that modeling the scene geometry with a Truncated Signed Distance Field (TSDF) leads to efficient and high-quality surface reconstruction. We verified through extensive experiments that our approach outperforms existing methods significantly in both reconstruction and localization accuracy while running up to one order of magnitude faster.

It should also be noted that ESLAM accepts and deals with the forgetting problem in exchange for memory preservation. Due to the structure of our feature plane representation, updating features to adapt to new geometry may affect previously reconstructed geometries. To address this issue, we keep track of previous keyframes and allocate a large portion of computation resources to retain and remember previously reconstructed regions. Although ESLAM is substantially faster than competing approaches, handling the forgetting problem more efficiently could further reduce frame processing time.

4.7 Supplementary Material

4.7.1 Further Implementation Details

This section provides additional implementation details of our method. For the sake of completeness, we also reiterate the points mentioned in Section 4.5.

The truncation distance T is set to 6 cm in our method. We employ coarse feature planes with a resolution of 24 cm for both geometry and appearance. For fine feature planes, we use a resolution of 6 cm for geometry and 3 cm for appearance. All feature planes have 32 channels, resulting in a 64-channel concatenated feature input for the decoders. The decoders are two-layer MLPs with 32 channels in the hidden layer. ReLU activation function is used for the hidden layer, and Tanh and Sigmoid are respectively used for the output layers of TSDF and raw colors.

We use different set of loss coefficients for mapping and tracking. During mapping we set $\lambda_{fs} = 5$, $\lambda_{T\text{-}m} = 200$, $\lambda_{T\text{-}t} = 10$, $\lambda_d = 0.1$, and $\lambda_c = 5$. And during tracking, we set $\lambda_{fs} = 10$, $\lambda_{T\text{-}m} = 200$, $\lambda_{T\text{-}t} = 50$, $\lambda_d = 1$, and $\lambda_c = 5$. These coefficients are obtained by performing grid search in our experiments.

For the scenes from Replica (Straub et al., 2019), we sample $N_{strat} = 32$ points for stratified sampling and $N_{imp} = 8$ points for importance sampling on each ray. We perform 15 optimization iterations for mapping and randomly select 4000 rays for each iteration. For camera tracking, 2000 rays are chosen at random and 8 optimization iterations are performed. Since ScanNet's (Dai et al., 2017) scenes are at a larger scale and more challenging, we set $N_{strat} = 48$ and $N_{imp} = 8$. Also, we perform 30 optimization iterations for both mapping and tracking in ScanNet's (Dai et al., 2017) scenes. For the scenes in TUM RGB-D dataset (Sturm et al., 2012), we similarly set $N_{strat} = 48$ and $N_{imp} = 8$. For this dataset, we perform 60 optimization iterations for mapping and 200 optimization iterations for tracking, and we randomly sample 5000 rays for each iteration.

We initiate the mapping process every 4 input frames and use a window of $W = 20$ keyframes for jointly optimizing the feature planes, MLP decoders, and camera poses of the selected keyframes. We use Adam (Kingma and Ba, 2014) for optimizing all learnable parameters of our method and set the learning rates according to a simple grid search in our experiments. We use a learning rate of 0.001 for the MLP decoders and a learning rate of 0.005 for the feature planes. We always use a learning rate of 0.001 for the camera poses, *i.e.*, rotation and translation $\{R, t\}$, of the selected keyframes during the joint optimization of the mapping step. During the tracking step in the Replica's (Straub et al., 2019) scenes, we use a learning rate of 0.001 for camera rotation and translation. For camera tracking in the scenes of ScanNet (Dai et al., 2017), we use a learning rate of 0.0005 for camera translation and a learning rate of 0.0025 for camera rotation. Lastly, For camera tracking in the scenes of TUM RGB-D (Sturm et al., 2012), we use a learning rate of 0.01 for camera translation and a learning rate of 0.002 for camera rotation. We model the camera rotation parameters with quaternions (Shoemake, 1985).

	Method	ATE↓	Acc.↓	Comp.↓
$\frac{1}{1}$ D	NICE-SLAM	1.69	1.71	1.69
	ESLAM (ours)	**0.71**	**1.07**	**1.12**
$\frac{1}{8}$ D	NICE-SLAM	2.01	2.18	1.98
	ESLAM (ours)	**0.72**	**1.16**	**1.23**

Table 4.5: Robustness to depth resolution comparison of our method with NICE-SLAM (Zhu et al., 2022) in terms of ATE RMSE (cm), reconstruction accuracy (cm), and reconstruction completion (cm) on `room0` of the Replica (Straub et al., 2019) dataset. Our method's accuracy is less affected when input depth is downsampled by a factor of $\frac{1}{8}$.

Once all input frames are processed, and for evaluation purposes, we build a TSDF volume for each scene and use the marching cubes algorithm (Lorensen and Cline, 1987) to obtain 3D meshes. We do not employ any post-processing for our representation or extracted meshes, except that, for quantitative evaluation, we cull faces from a mesh that are not inside any camera frustum or are occluded in all RGB-D frames. To ensure fairness, we do the same mesh culling before evaluating the previous approaches.

4.7.2 Ablation Study

In this section, we conduct various experiments to show the robustness of our method in different experimental settings and to validate our architecture design choices.

Robustness to Depth Quality. In this experiment, we evaluate how robust the methods are to the quality of input depths. Accordingly, we downsample input depths of `room0` of the replica dataset (Straub et al., 2019) to $\frac{1}{8}$ of the original resolution. The results in Table 4.5 reveal that our method's reconstruction and localization are less sensitive to the resolution of input depth maps.

Keyframe Policy. Whenever we perform a mapping step for an input frame, we always include that frame in our global keyframe list (see Section 4.4.4). NICE-SLAM (Zhu et al., 2022), on the other hand, only updates its keyframe list once per 10 mapping steps. To make sure that our evaluations are fair, we also run NICE-SLAM with our own keyframe updating policy on `room0` of the Replica (Straub et al., 2019) dataset. The results in Table 4.6 show that NICE-SLAM only slightly benefits from this updating policy.

Our Design Choices. We conduct multiple experiments in Table 4.7 to defend our design choices in ESLAM. These experiments are conducted on the scenes in the Replica (Straub et al., 2019) and ScanNet (Dai et al., 2017) datasets, and the details of the experimental settings are as follows. (a) We use shared feature planes for geometry and appearance (see Section 4.4.1 and Figure 4.2). (b) We only employ coarse feature planes (see Section 4.4.1 and Figure 4.2). (c) We only employ fine feature planes (see Section 4.4.1 and Figure 4.2). (d) We add the interpolated

Method	ATE↓	Acc.↓	Comp.↓
NICE-SLAM (Zhu et al., 2022)	1.69	1.71	1.69
NICE-SLAM w/ Our Key. Policy	1.65	1.68	1.66

Table 4.6: Analysis of the impact of our keyframe updating policy on NICE-SLAM (Zhu et al., 2022). The experiment is conducted on `room0` of Replica (Straub et al., 2019), and the metrics are ATE RMSE (cm), reconstruction accuracy (cm), and reconstruction completion (cm). NICE-SLAM only slightly benefits from our updating policy.

coarse $f^c_*(p_n)$ and fine $f^f_*(p_n)$ features instead of concatenating them (see Section 4.4.1 and Figure 4.2). (e) We discard importance sampling and use stratified sampling for all N points on a ray (see Section 4.4.2). (f) We only exploit depth inputs and discard color rendering and RGB inputs (see Section 4.4.2). (g) We do not consider separate loss functions for the points that are at the tail of the truncation region $P_r^{T\text{-}t}$ and for the points that are in the middle $P_r^{T\text{-}m}$ (see Section 4.4.3). (h) We do not jointly optimize camera poses during the mapping step (see Section 4.4.4). (i) We evaluate our full model. Note that due to the incompleteness of ScanNet's (Dai et al., 2017) ground truth meshes, we only evaluate localization accuracy on this dataset.

4.7.3 Additional Qualitative Analysis

This section provides additional qualitative analysis to contrast the capability of our method to preserve scene details in comparison to previous NeRF-based dense visual SLAM methods, iMAP* (Sucar et al., 2021) and NICE-SLAM (Zhu et al., 2022). We provide this analysis on the Replica dataset (Straub et al., 2019) in Figure 4.7 with both textured and untextured meshes. The results demonstrate that our method produces more accurate meshes with fewer artifacts.

4.7.4 Per-Scene Breakdown of the Results

In this section, we break down the quantitative analysis of Table 4.1 into a per-scene analysis. Tables 4.8 and 4.9 show the per-scene quantitative evaluation of our method in comparison with iMAP* (Sucar et al., 2021) and NICE-SLAM (Zhu et al., 2022) on the Replica dataset (Straub et al., 2019). As it is shown in Tables 4.8 and 4.9, our method outperforms previous approaches in all scenes of Replica (Straub et al., 2019). Also, lower variances in our experiments are an indication that our method is more stable from run to run.

4.7.5 Effect of Frame Processing Time

In this section, we investigate the trade-off between frame processing time and our method's reconstruction and localization accuracy. In this study, we increase the number of optimization iterations during the mapping and tracking. By default, our ESLAM method performs

Experiment	ScanNet	Replica		
	ATE↓	ATE↓	Acc.↓	Comp.↓
a. Using shared feature planes.	7.49	0.65	0.99	1.08
b. Using only the coarse feature planes.	7.53	0.97	1.12	1.29
c. Using only the fine planes.	8.27	0.72	1.00	1.09
d. Replacing the concatenation with summation.	7.55	0.64	0.98	1.07
e. No importance sampling.	7.44	0.67	1.08	1.14
f. No color rendering.	8.31	0.68	1.03	1.08
g. One loss for the whole truncation region.	8.28	0.71	1.01	1.10
h. No camera pose optimization in mapping.	11.27	4.85	2.23	2.21
i. Full ESLAM method.	**7.38**	**0.63**	**0.97**	**1.05**

Table 4.7: Ablation study of our design choices on the ScanNet (Dai et al., 2017) and Replica (Straub et al., 2019) datasets. The metrics are ATE RMSE (cm), reconstruction accuracy (cm), and reconstruction completion (cm). For the details of this study, see Section 4.7.2.

$Iter_m = 15$ optimization iterations during mapping and $Iter_t = 8$ optimization iterations during tracking for the scenes of the Replica dataset (Straub et al., 2019). We define ESLAM x2 as our method when we double the number of optimization iterations, *i.e.*, $Iter_m = 30$ and $Iter_t = 16$. And similarly, we define ESLAM x10 as our method with $Iter_m = 150$ and $Iter_t = 80$.

Table 4.10 provides a quantitative analysis of ESLAM x2 and ESLAM x10, as well as a comparison with our default ESLAM method and existing approaches. The results show that at the cost of increased frame processing time, our method yields more accurate scene reconstruction and camera trajectory. It should be noted that even ESLAM x10 runs faster than the existing state-of-the-art method, NICE-SLAM (Zhu et al., 2022).

Figure 4.8 provides a qualitative analysis of ESLAM x10 compared to our default ESLAM method. In this analysis, we render the scenes with untextured meshes to contrast the quality of geometry reconstruction. While the quality difference is subtle, Figure 4.8 indicates that increasing the number of optimization iterations results in more accurate geometry reconstruction and smoother surfaces.

	Methods	Reconstruction (cm)			
		Depth L1↓	Acc.↓	Comp.↓	Comp. Ratio (%)↑
room0	iMAP*	6.56 ± 0.39	5.89 ± 0.19	6.07 ± 0.22	66.55 ± 1.58
	NICE-SLAM	2.77 ± 0.13	1.71 ± 0.03	1.69 ± 0.03	97.61 ± 0.09
	ESLAM (Ours)	**0.97 ± 0.04**	**1.07 ± 0.01**	**1.12 ± 0.01**	**99.06 ± 0.05**
room1	iMAP*	5.97 ± 1.14	5.71 ± 0.31	5.57 ± 0.40	66.04 ± 3.45
	NICE-SLAM	2.52 ± 0.11	1.36 ± 0.03	1.34 ± 0.04	98.60 ± 0.14
	ESLAM (Ours)	**1.07 ± 0.07**	**0.85 ± 0.01**	**0.88 ± 0.01**	**99.64 ± 0.06**
room2	iMAP*	7.82 ± 0.94	6.34 ± 0.32	5.47 ± 0.27	69.87 ± 4.15
	NICE-SLAM	3.54 ± 0.35	1.75 ± 0.06	1.71 ± 0.03	96.52 ± 0.26
	ESLAM (Ours)	**1.28 ± 0.07**	**0.93 ± 0.01**	**1.05 ± 0.01**	**98.84 ± 0.06**
office0	iMAP*	7.57 ± 0.70	7.44 ± 0.26	5.13 ± 0.37	70.97 ± 3.52
	NICE-SLAM	2.17 ± 0.14	1.43 ± 0.06	1.56 ± 0.05	96.30 ± 0.33
	ESLAM (Ours)	**0.86 ± 0.02**	**0.85 ± 0.01**	**0.96 ± 0.01**	**98.34 ± 0.05**
office1	iMAP*	8.91 ± 0.65	10.34 ± 0.15	5.58 ± 0.24	72.08 ± 3.21
	NICE-SLAM	2.41 ± 0.11	1.16 ± 0.07	1.15 ± 0.03	98.04 ± 0.19
	ESLAM (Ours)	**1.26 ± 0.02**	**0.83 ± 0.06**	**0.81 ± 0.01**	**98.85 ± 0.08**
office2	iMAP*	11.04 ± 0.69	9.15 ± 0.39	6.27 ± 0.37	62.24 ± 2.62
	NICE-SLAM	4.96 ± 0.58	1.83 ± 0.07	1.72 ± 0.03	96.96 ± 0.25
	ESLAM (Ours)	**1.71 ± 0.07**	**1.02 ± 0.01**	**1.09 ± 0.01**	**98.60 ± 0.12**
office3	iMAP*	10.12 ± 1.31	7.14 ± 0.27	6.02 ± 0.20	66.07 ± 1.65
	NICE-SLAM	4.91 ± 0.70	2.24 ± 0.17	2.17 ± 0.05	93.08 ± 0.40
	ESLAM (Ours)	**1.43 ± 0.05**	**1.21 ± 0.01**	**1.42 ± 0.01**	**96.80 ± 0.03**
office4	iMAP*	7.85 ± 1.32	5.32 ± 0.18	6.51 ± 0.20	63.63 ± 1.39
	NICE-SLAM	3.81 ± 0.74	2.09 ± 0.16	2.03 ± 0.17	95.00 ± 1.31
	ESLAM (Ours)	**1.06 ± 0.08**	**1.15 ± 0.02**	**1.27 ± 0.01**	**97.65 ± 0.14**
Average	iMAP*	8.23 ± 0.88	7.16 ± 0.26	5.83 ± 0.27	67.17 ± 2.70
	NICE-SLAM	3.29 ± 0.33	1.66 ± 0.07	1.63 ± 0.05	96.74 ± 0.36
	ESLAM (Ours)	**1.18 ± 0.05**	**0.97 ± 0.02**	**1.05 ± 0.01**	**98.60 ± 0.07**

Table 4.8: Per-scene quantitative comparison of our proposed ESLAM with existing NeRF-based dense visual SLAM models, iMAP* (Sucar et al., 2021) and NICE-SLAM (Zhu et al., 2022), on the Replica dataset (Straub et al., 2019) in terms of reconstruction metrics. The results are the average and standard deviation of five independent runs on each scene of the Replica dataset. Our method outperforms previous works by a high margin and has lower variances, indicating it is also more stable from run to run. The evaluation metrics for reconstruction are L1 loss (cm) between rendered and ground truth depth maps of 1000 random camera poses, reconstruction accuracy (cm), reconstruction completion (cm), and completion ratio (%). It should also be noted that our method runs up to ×10 faster on this dataset (see Section 4.5.2 for runtime analysis).

	Methods	Localization (cm)	
		ATE Mean↓	ATE RMSE↓
room0	iMAP*	3.12 ± 0.84	5.23 ± 1.41
	NICE-SLAM	1.43 ± 0.09	1.69 ± 0.17
	ESLAM (Ours)	**0.61** ± **0.06**	**0.71** ± **0.13**
room1	iMAP*	2.54 ± 0.37	3.09 ± 0.48
	NICE-SLAM	1.70 ± 0.29	2.13 ± 0.24
	ESLAM (Ours)	**0.56** ± **0.02**	**0.70** ± **0.02**
room2	iMAP*	2.31 ± 0.20	2.58 ± 0.19
	NICE-SLAM	1.41 ± 0.24	1.87 ± 0.39
	ESLAM (Ours)	**0.43** ± **0.01**	**0.52** ± **0.01**
office0	iMAP*	1.69 ± 1.06	2.40 ± 1.05
	NICE-SLAM	1.12 ± 0.22	1.26 ± 0.24
	ESLAM (Ours)	**0.42** ± **0.03**	**0.57** ± **0.04**
office1	iMAP*	1.03 ± 0.17	1.17 ± 0.25
	NICE-SLAM	0.74 ± 0.19	0.84 ± 0.17
	ESLAM (Ours)	**0.46** ± **0.05**	**0.55** ± **0.04**
office2	iMAP*	3.99 ± 0.98	5.67 ± 1.82
	NICE-SLAM	1.42 ± 0.10	1.71 ± 0.14
	ESLAM (Ours)	**0.47** ± **0.03**	**0.58** ± **0.09**
office3	iMAP*	4.05 ± 0.93	5.08 ± 1.37
	NICE-SLAM	2.31 ± 0.51	3.98 ± 1.79
	ESLAM (Ours)	**0.61** ± **0.03**	**0.72** ± **0.02**
office4	iMAP*	1.93 ± 0.21	2.23 ± 0.35
	NICE-SLAM	2.22 ± 0.68	2.82 ± 0.71
	ESLAM (Ours)	**0.52** ± **0.02**	**0.63** ± **0.03**
Average	iMAP*	2.59 ± 0.58	3.42 ± 0.87
	NICE-SLAM	1.56 ± 0.29	2.05 ± 0.45
	ESLAM (Ours)	**0.52** ± **0.03**	**0.63** ± **0.05**

Table 4.9: Per-scene quantitative comparison of our proposed ESLAM with existing NeRF-based dense visual SLAM models, iMAP* (Sucar et al., 2021) and NICE-SLAM (Zhu et al., 2022), on the Replica dataset (Straub et al., 2019) in terms of localization accuracy. The results are the average and standard deviation of five independent runs on each scene of the Replica dataset. Our method outperforms previous works by a high margin and has lower variances, indicating it is also more stable from run to run. The evaluation metrics for localization are mean and RMSE of ATE (cm) (Sturm et al., 2012). It should also be noted that our method runs up to ×10 faster on this dataset (see Section 4.5.2 for runtime analysis).

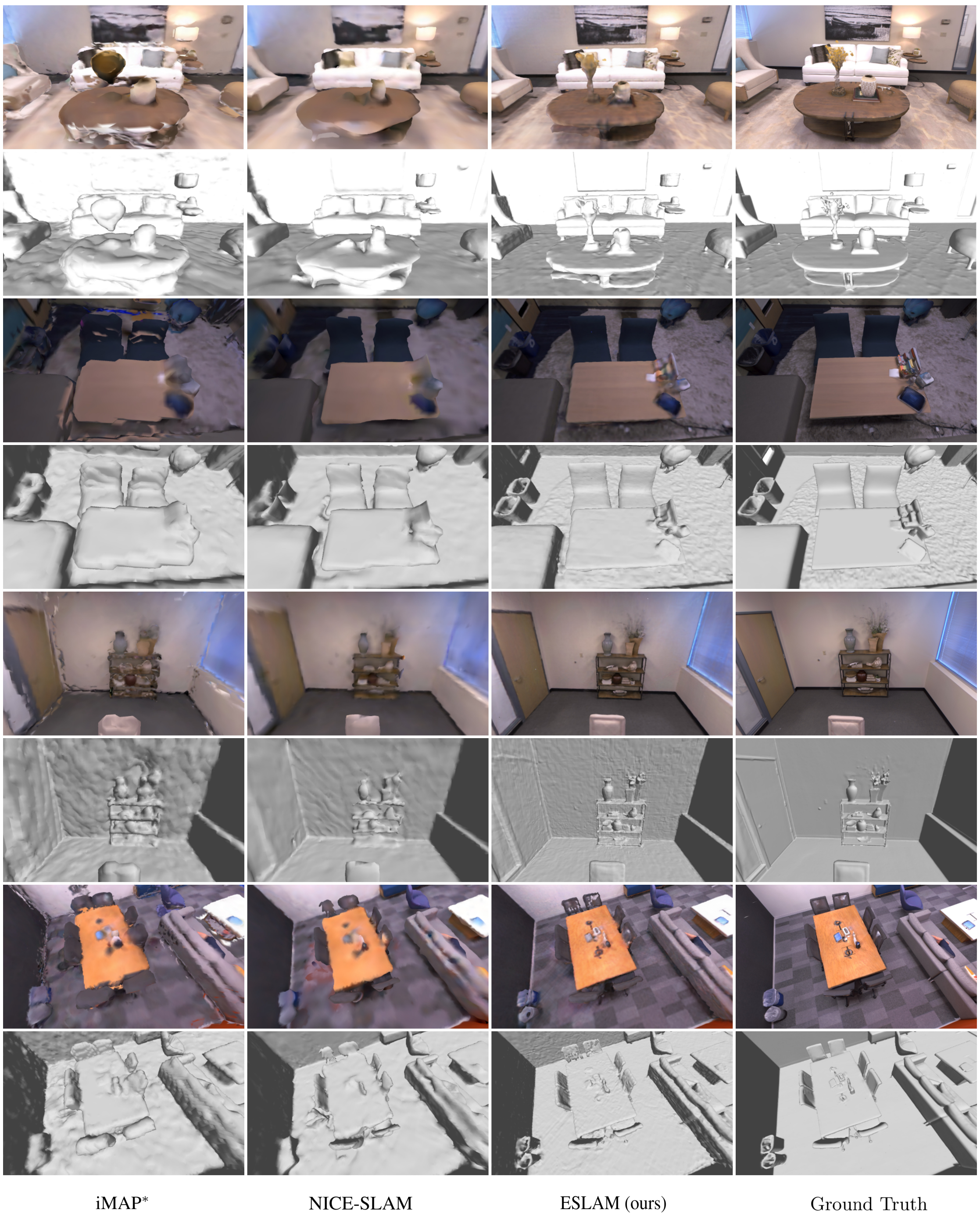


Figure 4.7: Qualitative comparison of our method's scene reconstruction with that of iMAP* (Sucar et al., 2021) and NICE-SLAM (Zhu et al., 2022) on the Replica dataset (Straub et al., 2019). Our method produces more accurate detailed geometry as well as higher-quality textures. The scenes are rendered with both textured and untextured meshes and the ground truth textured images are rendered with the ReplicaViewer software (Straub et al., 2019). It should also be noted that our method runs up to ×10 faster on this dataset (see Section 4.5.2 for runtime analysis).

Method	Optimization Iterations	Acc.↓	Comp.↓	ATE↓	FPT↓
iMAP*	-	7.16	5.83	3.42	5.20
NICE-SLAM	-	1.66	1.63	2.05	2.10
ESLAM (ours)	$Iter_m = 15, \quad Iter_t = 8$	0.97	1.05	0.63	**0.18**
ESLAM x2 (ours)	$Iter_m = 30, \quad Iter_t = 16$	0.95	1.03	0.42	0.35
ESLAM x10 (ours)	$Iter_m = 150, Iter_t = 80$	**0.92**	**1.01**	**0.31**	1.72

Table 4.10: Quantitative analysis of the effect of the number of optimization iterations during mapping and tracking on our method's reconstruction and localization accuracy. $Iter_m$ stands for the number of optimization iterations during mapping, and $Iter_t$ denotes the number of optimization iterations during tracking. The evaluation metrics are reconstruction accuracy (cm), reconstruction completion (cm), and ATE RMSE (cm) (Sturm et al., 2012). Average Frame Processing Time (FTP) in seconds is also shown to highlight the trade-off between the accuracy and throughput of our method. For reference, we reiterate the performance of the existing approaches, iMAP* (Sucar et al., 2021) and NICE-SLAM (Zhu et al., 2022). It should be noted that even ESLAM x10 runs faster than the existing state-of-the-art method, NICE-SLAM (Zhu et al., 2022). Refer to Section 4.7.5 for the details of this experiment, and see Figure 4.8 for the qualitative analysis.

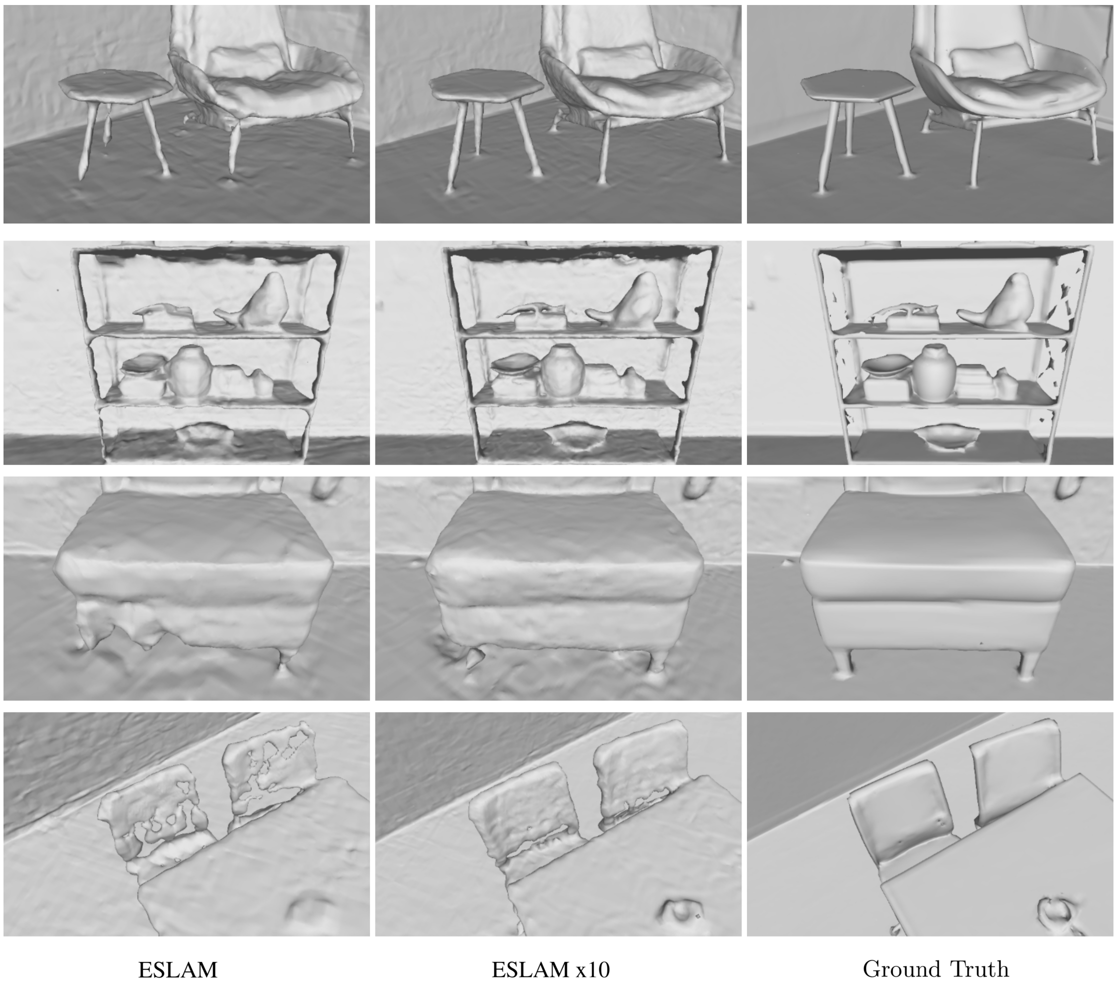


Figure 4.8: Qualitative analysis of the effect of the number of optimization iterations during mapping and tracking on our method's reconstruction quality. ESLAM x10 is our method when we multiply the number of optimization iterations by 10. Refer to Section 4.7.5 for the details of this experiment, and see Table 4.10 for the quantitative analysis.

Conclusion Part III

5 Conclusion and Future Work

5.1 Conclusion

In this thesis, we have made significant contributions to the fields of 3D computer vision and scene understanding through the introduction of three novel approaches. Firstly, we propose a novel approach to improve the accuracy of monocular depth estimation from depth sensor raw data. Subsequently, we introduce a new generalizable method that enhances the quality of synthesized images from novel camera poses. Finally, we introduce an efficient dense visual SLAM system that outperforms state-of-the-art methods in both accuracy and efficiency.

We have presented DepthInSpace in Chapter 2 which represents a significant leap forward in the accurate estimation of depth from structured-light sensor data. By integrating optical flow and harnessing information from multiple video frames within a self-supervised framework, DepthInSpace enhances depth estimation accuracy and outperforms existing methods, as evidenced by qualitative and quantitative evaluations across diverse datasets, including synthetic and real-world scenes.

Moving on to our second contribution in Chapter 3, GeoNeRF stands out as a pioneering generalizable method for novel view synthesis. This approach not only achieves state-of-the-art image quality for complex scenes but also eliminates the need for per-scene optimization. Leveraging recent advancements in multi-view architectures and radiance fields, GeoNeRF constructs cascaded cost volumes for source views, which are then aggregated through an attention-based network for synthesizing images from novel poses. Furthermore, we propose promising avenues for future research, suggesting that the incorporation of advanced algorithms to dynamically select nearby views or an adaptive approximation of the optimal number of required cost volumes could further enhance the versatility and efficiency of GeoNeRF.

Lastly, our third contribution in Chapter 4, ESLAM, introduces a top-tier dense visual SLAM by leveraging Neural Radiance Fields to improve both speed and accuracy. Through the innovative replacement of voxel grid representation with axis-aligned feature planes and the adoption of a Truncated Signed Distance Field for scene geometry modeling, ESLAM showcases remarkable advancements in reconstruction and localization accuracy. Notably, our experiments validate that ESLAM improves existing methods' accuracy significantly while running up to one order of magnitude faster.

Collectively, these three contributions not only advance the state of the art in their respective domains but also set the stage for further research and applications, demonstrating the transformative potential of our innovative approaches in the broader landscape of 3D computer vision research.

5.2 Future Work and Recent Advances in Successor Studies

Considering the advancements highlighted in this thesis, numerous exciting opportunities for future research emerge within the domains of depth estimation, novel view synthesis, and dense visual SLAM. It is worth noting that due to the rapid expansion of research in these areas, there are already several concurrent or inspired works documented in the literature as of the time of this writing. This section offers a concise overview of potential extensions, both existing and prospective.

To enhance **DepthInSpace**, there is potential for additional investigation into expanding the self-supervised framework to include active stereo. This involves utilizing a stereo camera to capture the illuminated pattern, which contributes supplementary information for the algorithm to reason about geometry and occlusion. Notably, this approach could allow for the adoption of a sparser projection pattern, offering advantages in terms of power consumption and efficiency.

An additional avenue of research involves addressing the constraints imposed by the current dataset. Despite our exploration of various synthetic datasets and the evaluation of our method on real-world scenes, the absence of an extensive real dataset remains a limiting factor for the effectiveness of our proposed approach. A potential strategy to alleviate this limitation is to adapt the model to leverage recent advancements in conventional monocular depth estimation. Utilizing large datasets, such as those used by Ranftl et al. (2020), in this domain could provide robust prior information for the network, helping to mitigate the aforementioned issue to some extent.

An alternative enhancement to boost the generalizability of DepthInSpace to out-of-distribution depths involves substituting the single-frame depth regressor network with a cost-volume-based depth estimator network, such as MVSNet (Yao et al., 2018). This modification makes the network indifferent to the absolute values of depths or disparities. Consequently, the network becomes capable of functioning on novel datasets where depth values exhibit a significant shift compared to those in the training sets. Such architecture is employed in the concurrent work by Li et al. (2023b), and a similar idea is explored in another coexisting work, GigaDepth (Schreiberhuber et al., 2022). GigaDepth employs a hierarchy of adaptive MLPs to robustly predict depth instead of opting for direct regression through a CNN.

Lastly, exploring the application of structured-light for depth estimation in dynamic scenes represents a natural extension for future research. This introduces novel challenges in comprehending and modeling temporal changes within the scene. Recent concurrent studies conducted by Qiao et al. (2022) and Qiao et al. (2023) have specifically attempted to tackle these challenges.

In our **GeoNeRF** contribution, a promising avenue for future research involves enhancing efficiency and minimizing computational resource requirements. Despite achieving exceptional quality in the rendered output, GeoNeRF does not rank among the most efficient rendering algorithms. Notably, the attention blocks in the renderer section of GeoNeRF stand out as the most computationally intensive aspect of the algorithm.

To alleviate the computational burden in these blocks, one potential modification involves adaptively filtering out less pertinent input source views. This strategy aims to decrease the number of tokens processed per pixel. Additionally, implementing more efficient transformer attention architectures, such as Perceiver IO (Jaegle et al., 2021) or SegFormer (Xie et al., 2021), could further diminish the computation costs associated with the renderer.

Another research direction to speed up GeoNeRF could be reducing the number of cast rays. More specifically, rendering a low-resolution image using anti-aliasing radiance field approaches (Barron et al., 2021; Hu et al., 2023), and scaling up the image using a lightweight super-resolution network can significantly reduce the computation costs. Exploiting such a technique along with feature rendering instead of RGB rendering has shown promising progress in the NeRF literature (Huang et al., 2023b; Wang et al., 2022c; Han et al., 2024).

Concurrent or after our research in GeoNeRF, the following studies also explored the field of generalizable 3D understanding. WaveNeRF (Xu et al., 2023b) integrates wavelet frequency decomposition into MVS and NeRF to achieve generalizable yet high-quality synthesis without any per-scene optimization. NeuRay (Liu et al., 2022) exploits MVS to detect occluded source views and proposes an implicit occlusion-aware feature aggregation. Du et al. (2023) and Suhail et al. (2022) eliminate alpha compositing in ray casting by using an attention-based aggregation of the image-based features and directly regressing the output color to save computation. LIRF (Huang et al., 2023c) supports generalizable NeRF rendering at arbitrary scales by proposing an anti-aliasing architecture. Liu et al. (2024) and (Liu et al., 2023) introduce a method that generates a 3D textured mesh from a single image exploiting diffusion models. And ContraNeRF (Yang et al., 2023a) explores synthetic-to-real generalization of radiance fields.

Lastly, while our **ESLAM** approach has made significant strides in enhancing the accuracy and efficiency of implicit visual SLAM, it still falls short of the real-time performance and precise localization capabilities of traditional SLAM methods. Specifically, techniques like loop closure detection, global bundle adjustment, and second-order optimization of camera poses, which have proven their effectiveness in traditional SLAM, need to be incorporated and tailored to the unique framework of implicit neural-based visual SLAM systems like ours. In successor literature, MIPS-Fusion (Tang et al., 2023b) and NGEL-SLAM (Mao et al., 2023) propose to learn implicit sub-maps for the environment and introduce submap-level loop closure detection and correction. Haghighi et al. (2023) completely decouple tracking and mapping representation and use the traditional ORB-SLAM 3 (Campos et al., 2021) as a tracking backbone.

Another compelling extension to our research in ESLAM would be to investigate techniques for reducing the computational burden associated with addressing the forgetting problem. Our current representation, heavily reliant on feature planes, is particularly susceptible to this challenge. A promising strategy to tackle this issue lies in leveraging recent advancements in the field of continual learning. Techniques such as Elastic Weight Consolidation (EWC) (Kirkpatrick et al., 2017), which preserve the knowledge of old tasks while learning new ones, could be seamlessly integrated into our ESLAM optimization scheme, effectively alleviating the need to re-sample pixels from previous key-frames.

Concurrent or after our work in ESLAM, the following works also explored the field of implicit representation in visual SLAM. Vox-Fusion (Yang et al., 2022a) and Point-SLAM (Sandström et al., 2023) address the memory constraints of previous methods by employing a sparse octree structure or a sparse point cloud, respectively, for the 3D map. DNS SLAM (Li et al., 2023a) and SNI-SLAM (Zhu et al., 2023a) incorporate semantic information in the SLAM pipeline. NID-SLAM (Xu et al., 2024) and DDN-SLAM (Li et al., 2024) extend the visual SLAM to dynamic scenes. Liu and Zhu (2023) introduces map fusion of multiple SLAM agents. NICER-SLAM (Zhu et al., 2023b) and HI-SLAM (Zhang et al., 2023a) investigate dense mapping in a monocular RGB SLAM setting while EN-SLAM (Qu et al., 2023) integrates event sensor data with the RGBD SLAM system. More recently, with the emergence of Gaussian Splatting (Kerbl et al., 2023), there has been a growing interest in integrating this novel efficient 3D representation into the dense visual SLAM pipeline (Matsuki et al., 2023; Yan et al., 2023; Yugay et al., 2023; Keetha et al., 2023; Huang et al., 2023a).

Overall, our contributions have opened several new avenues for research in 3D computer vision. By leveraging our novel techniques and insights, researchers can develop more efficient and performant algorithms for tasks such as depth estimation, novel view synthesis, 3D reconstruction, scene understanding, and SLAM. These advancements have the potential to broaden the reach and impact of 3D computer vision across various domains, from robotics and augmented reality to autonomous driving and medical imaging.

Bibliography

Aliev, K.-A., Sevastopolsky, A., Kolos, M., Ulyanov, D., and Lempitsky, V. (2020). 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.

Arandjelović, R. and Zisserman, A. (2021). Nerf in detail: Learning to sample for view synthesis. *arXiv preprint arXiv:2106.05264*.

Azinović, D., Martin-Brualla, R., Goldman, D. B., Nießner, M., and Thies, J. (2022). Neural rgb-d surface reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6290–6301.

Barron, J. T., Mildenhall, B., Tancik, M., Hedman, P., Martin-Brualla, R., and Srinivasan, P. P. (2021). Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5855–5864.

Bartoccioni, F., Zablocki, É., Pérez, P., Cord, M., and Alahari, K. (2023). Lidartouch: Monocular metric depth estimation with a few-beam lidar. *Computer Vision and Image Understanding*, 227:103601.

Bernardini, F., Mittleman, J., Rushmeier, H., Silva, C., and Taubin, G. (1999). The ball-pivoting algorithm for surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 5(4):349–359.

Bian, J., Li, Z., Wang, N., Zhan, H., Shen, C., Cheng, M.-M., and Reid, I. (2019). Unsupervised scale-consistent depth and ego-motion learning from monocular video. In *Advances in neural information processing systems*, pages 35–45.

Bloesch, M., Czarnowski, J., Clark, R., Leutenegger, S., and Davison, A. J. (2018). Codeslam—learning a compact, optimisable representation for dense visual slam. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2560–2568.

Bloesch, M., Omari, S., Hutter, M., and Siegwart, R. (2015). Robust visual inertial odometry using a direct ekf-based approach. In *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 298–304. IEEE.

Boulch, A. (2020). Convpoint: Continuous convolutions for point cloud processing. *Computers & Graphics*.

Bozic, A., Palafox, P., Thies, J., Dai, A., and Nießner, M. (2021). Transformerfusion: Monocular rgb scene reconstruction using transformers. *Advances in Neural Information Processing Systems*, 34:1403–1414.

Buehler, C., Bosse, M., McMillan, L., Gortler, S., and Cohen, M. (2001). Unstructured lumigraph rendering. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 425–432.

Butler, D. J., Wulff, J., Stanley, G. B., and Black, M. J. (2012). A naturalistic open source movie for optical flow evaluation. In *European conference on computer vision*, pages 611–625. Springer.

Campos, C., Elvira, R., Rodríguez, J. J. G., Montiel, J. M., and Tardós, J. D. (2021). Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam. *IEEE Transactions on Robotics*, 37(6):1874–1890.

Cao, A.-Q. and de Charette, R. (2023). Scenerf: Self-supervised monocular 3d scene reconstruction with radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9387–9398.

Casas, D., Richardt, C., Collomosse, J., Theobalt, C., and Hilton, A. (2015). 4d model flow: Precomputed appearance alignment for real-time 4d video interpolation. In *Computer Graphics Forum*, volume 34, pages 173–182. Wiley Online Library.

Casser, V., Pirk, S., Mahjourian, R., and Angelova, A. (2019). Unsupervised monocular depth and ego-motion learning with structure and semantics. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*.

Chan, E. R., Lin, C. Z., Chan, M. A., Nagano, K., Pan, B., De Mello, S., Gallo, O., Guibas, L. J., Tremblay, J., Khamis, S., et al. (2022). Efficient geometry-aware 3d generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16123–16133.

Chang, A. X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., Xiao, J., Yi, L., and Yu, F. (2015). ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago.

Chen, A., Xu, Z., Geiger, A., Yu, J., and Su, H. (2022). Tensorf: Tensorial radiance fields. In *European Conference on Computer Vision (ECCV)*.

Chen, A., Xu, Z., Zhao, F., Zhang, X., Xiang, F., Yu, J., and Su, H. (2021). Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14124–14133.

Chen, Q. and Koltun, V. (2014). Fast mrf optimization with application to depth reconstruction. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3914–3921.

Chen, Y. and Medioni, G. (1992). Object modelling by registration of multiple range images. *Image Vis. Comput.*, 10:145–155.

Chen, Y., Yang, B., Liang, M., and Urtasun, R. (2019). Learning joint 2d-3d representations for depth completion. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 10023–10032.

Chen, Z., Wang, F., and Liu, H. (2023). Text-to-3d using gaussian splatting. *arXiv preprint arXiv:2309.16585*.

Cheng, S., Xu, Z., Zhu, S., Li, Z., Li, L. E., Ramamoorthi, R., and Su, H. (2020). Deep stereo using adaptive thin volume representation with uncertainty awareness. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2524–2534.

Chib, P. S. and Singh, P. (2023). Recent advancements in end-to-end autonomous driving using deep learning: A survey. *IEEE Transactions on Intelligent Vehicles*.

Chibane, J., Bansal, A., Lazova, V., and Pons-Moll, G. (2021). Stereo radiance fields (srf): Learning view synthesis for sparse views of novel scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7911–7920.

Choe, J., Im, S., Rameau, F., Kang, M., and Kweon, I. S. (2021). Volumefusion: Deep depth fusion for 3d scene reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16086–16095.

Chung, C.-M., Tseng, Y.-C., Hsu, Y.-C., Shi, X.-Q., Hua, Y.-H., Yeh, J.-F., Chen, W.-C., Chen, Y.-T., and Hsu, W. H. (2023). Orbeez-slam: A real-time monocular visual slam with orb features and nerf-realized mapping. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9400–9406. IEEE.

Czarnowski, J., Laidlow, T., Clark, R., and Davison, A. J. (2020). Deepfactors: Real-time probabilistic dense monocular slam. *IEEE Robotics and Automation Letters*, 5(2):721–728.

Dai, A., Chang, A. X., Savva, M., Halber, M., Funkhouser, T., and Nießner, M. (2017). Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5828–5839.

De Bonet, J. S. and Viola, P. (1999). Poxels: Probabilistic voxelized volume reconstruction. In *Proceedings of International Conference on Computer Vision (ICCV)*, pages 418–425.

Debevec, P., Yu, Y., and Borshukov, G. (1998). Efficient view-dependent image-based rendering with projective texture-mapping. In *Eurographics Workshop on Rendering Techniques*, pages 105–116. Springer.

Debevec, P. E., Taylor, C. J., and Malik, J. (1996). Modeling and rendering architecture from photographs: A hybrid geometry-and image-based approach. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 11–20.

Deng, K., Liu, A., Zhu, J.-Y., and Ramanan, D. (2022). Depth-supervised nerf: Fewer views and faster training for free. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12882–12891.

DeVries, T., Bautista, M. A., Srivastava, N., Taylor, G. W., and Susskind, J. M. (2021). Unconstrained scene generation with locally conditioned radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14304–14313.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. (2021). An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*.

Du, R., Chuang, M., Chang, W., Hoppe, H., and Varshney, A. (2018). Montage4d: interactive seamless fusion of multiview video textures. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, pages 1–11.

Du, Y., Smith, C., Tewari, A., and Sitzmann, V. (2023). Learning to render novel views from wide-baseline stereo pairs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4970–4980.

Endres, F., Hess, J., Sturm, J., Cremers, D., and Burgard, W. (2013). 3-d mapping with an rgb-d camera. *IEEE transactions on robotics*, 30(1):177–187.

Engel, J., Koltun, V., and Cremers, D. (2017). Direct sparse odometry. *IEEE transactions on pattern analysis and machine intelligence*, 40(3):611–625.

Engel, J., Schöps, T., and Cremers, D. (2014). Lsd-slam: Large-scale direct monocular slam. In *European conference on computer vision*, pages 834–849. Springer.

Fanello, S. R., Valentin, J., Rhemann, C., Kowdle, A., Tankovich, V., Davidson, P., and Izadi, S. (2017). Ultrastereo: Efficient learning-based matching for active stereo systems. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6535–6544. IEEE.

Flynn, J., Broxton, M., Debevec, P., DuVall, M., Fyffe, G., Overbeck, R., Snavely, N., and Tucker, R. (2019). Deepview: View synthesis with learned gradient descent. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2367–2376.

Flynn, J., Neulander, I., Philbin, J., and Snavely, N. (2016). Deepstereo: Learning to predict new views from the world's imagery. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5515–5524.

Forster, C., Pizzoli, M., and Scaramuzza, D. (2014). Svo: Fast semi-direct monocular visual odometry. In *2014 IEEE international conference on robotics and automation (ICRA)*, pages 15–22. IEEE.

Fridovich-Keil, S., Yu, A., Tancik, M., Chen, Q., Recht, B., and Kanazawa, A. (2022). Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5501–5510.

Furukawa, Y. and Ponce, J. (2009). Accurate, dense, and robust multiview stereopsis. *IEEE transactions on pattern analysis and machine intelligence*, 32(8):1362–1376.

Gao, C., Saraf, A., Kopf, J., and Huang, J.-B. (2021). Dynamic view synthesis from dynamic monocular video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5712–5721.

Gao, J., Gu, C., Lin, Y., Zhu, H., Cao, X., Zhang, L., and Yao, Y. (2023). Relightable 3d gaussian: Real-time point cloud relighting with brdf decomposition and ray tracing. *arXiv preprint arXiv:2311.16043*.

Garbin, S. J., Kowalski, M., Johnson, M., Shotton, J., and Valentin, J. (2021). Fastnerf: High-fidelity neural rendering at 200fps. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14346–14355.

Ghasemi, Y., Jeong, H., Choi, S. H., Park, K.-B., and Lee, J. Y. (2022). Deep learning-based object detection in augmented reality: A systematic review. *Computers in Industry*, 139:103661.

Godard, C., Mac Aodha, O., and Brostow, G. J. (2017). Unsupervised monocular depth estimation with left-right consistency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 270–279.

Godard, C., Mac Aodha, O., Firman, M., and Brostow, G. J. (2019). Digging into self-supervised monocular depth estimation. In *Proceedings of the IEEE international conference on computer vision*, pages 3828–3838.

Gu, X., Fan, Z., Zhu, S., Dai, Z., Tan, F., and Tan, P. (2020). Cascade cost volume for high-resolution multi-view stereo and stereo matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2495–2504.

Guizilini, V., Ambrus, R., Pillai, S., Raventos, A., and Gaidon, A. (2020). 3d packing for self-supervised monocular depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2485–2494.

Guo, H., Peng, S., Lin, H., Wang, Q., Zhang, G., Bao, H., and Zhou, X. (2022). Neural 3d scene reconstruction with the manhattan-world assumption. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5511–5520.

Hafner, D., Demetz, O., and Weickert, J. (2013). Why is the census transform good for robust optic flow computation? In *International Conference on Scale Space and Variational Methods in Computer Vision*, pages 210–221. Springer.

Haghighi, Y., Kumar, S., Thiran, J. P., and Van Gool, L. (2023). Neural implicit dense semantic slam. *arXiv preprint arXiv:2304.14560*.

Han, K., Xiang, W., and Yu, L. (2024). Volume feature rendering for fast neural radiance field reconstruction. *Advances in Neural Information Processing Systems*, 36.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Henzler, P., Mitra, N. J., and Ritschel, T. (2020). Learning a neural 3d texture space from 2d exemplars. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8356–8364.

Hirschmuller, H. (2007). Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on pattern analysis and machine intelligence*, 30(2):328–341.

Ho, C.-J., Tai, C.-H., Lin, Y.-Y., Yang, M.-H., and Tsai, Y.-H. (2023). Diffusion-ss3d: Diffusion model for semi-supervised 3d object detection. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Hu, W., Wang, Y., Ma, L., Yang, B., Gao, L., Liu, X., and Ma, Y. (2023). Tri-miprf: Tri-mip representation for efficient anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19774–19783.

Huang, H., Li, L., Cheng, H., and Yeung, S.-K. (2023a). Photo-slam: Real-time simultaneous localization and photorealistic mapping for monocular, stereo, and rgb-d cameras. *arXiv preprint arXiv:2311.16728*.

Huang, J., Thies, J., Dai, A., Kundu, A., Jiang, C., Guibas, L. J., Nießner, M., Funkhouser, T., et al. (2020). Adversarial texture optimization from rgb-d scans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1559–1568.

Huang, P.-H., Matzen, K., Kopf, J., Ahuja, N., and Huang, J.-B. (2018). Deepmvs: Learning multi-view stereopsis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2821–2830.

Huang, X., Li, W., Hu, J., Chen, H., and Wang, Y. (2023b). Refsr-nerf: Towards high fidelity and super resolution view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8244–8253.

Huang, X., Zhang, Q., Feng, Y., Li, X., Wang, X., and Wang, Q. (2023c). Local implicit ray function for generalizable radiance field representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 97–107.

Hui, T.-W., Tang, X., and Change Loy, C. (2018). Liteflownet: A lightweight convolutional neural network for optical flow estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8981–8989.

Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., and Brox, T. (2017). Flownet 2.0: Evolution of optical flow estimation with deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2462–2470.

Im, S., Jeon, H.-G., Lin, S., and Kweon, I. S. (2018). Dpsnet: End-to-end deep plane sweep stereo. In *International Conference on Learning Representations*.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456.

Jadhav, D., Chavan, G., Bagal, V., and Manza, R. (2022). Review on multimodal biometric recognition system using machine learning. In *Artificial Intelligence and Applications*.

Jaegle, A., Borgeaud, S., Alayrac, J.-B., Doersch, C., Ionescu, C., Ding, D., Koppula, S., Zoran, D., Brock, A., Shelhamer, E., et al. (2021). Perceiver io: A general architecture for structured inputs & outputs. *arXiv preprint arXiv:2107.14795*.

Jain, A., Tancik, M., and Abbeel, P. (2021). Putting nerf on a diet: Semantically consistent few-shot view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5885–5894.

Jancosek, M. and Pajdla, T. (2011). Multi-view reconstruction preserving weakly-supported surfaces. In *CVPR 2011*, pages 3121–3128. IEEE.

Jensen, R., Dahl, A., Vogiatzis, G., Tola, E., and Aanæs, H. (2014). Large scale multi-view stereopsis evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 406–413.

Jeong, Y., Ahn, S., Choy, C., Anandkumar, A., Cho, M., and Park, J. (2021). Self-calibrating neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5846–5854.

Jiang, S., Campbell, D., Lu, Y., Li, H., and Hartley, R. (2021). Learning to estimate hidden motions with global motion aggregation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Johari, M. M., Carta, C., and Fleuret, F. (2021). Depthinspace: Exploitation and fusion of multiple video frames for structured-light depth estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6039–6048.

Johari, M. M., Carta, C., and Fleuret, F. (2023). Eslam: Efficient dense slam system based on hybrid representation of signed distance fields. In *Proceedings of the IEEE/CVF Conference* *on Computer Vision and Pattern Recognition (CVPR)*, pages 17408–17419. Selected as a **Highlight Paper** (Top 2.5%).

Johari, M. M., Lepoittevin, Y., and Fleuret, F. (2022). Geonerf: Generalizing nerf with geometry priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18365–18375.

Kalantari, N. K., Wang, T.-C., and Ramamoorthi, R. (2016). Learning-based view synthesis for light field cameras. *ACM Transactions on Graphics (TOG)*, 35(6):1–10.

Kaya, B., Kumar, S., Sarno, F., Ferrari, V., and Van Gool, L. (2022). Neural radiance fields approach to deep multi-view photometric stereo. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1965–1977.

Keetha, N., Karhade, J., Jatavallabhula, K. M., Yang, G., Scherer, S., Ramanan, D., and Luiten, J. (2023). Splatam: Splat, track & map 3d gaussians for dense rgb-d slam. *arXiv preprint arXiv:2312.02126*.

Kerbl, B., Kopanas, G., Leimkühler, T., and Drettakis, G. (2023). 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4).

Kerl, C., Sturm, J., and Cremers, D. (2013). Dense visual slam for rgb-d cameras. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2100–2106. IEEE.

Keselman, L., Iselin Woodfill, J., Grunnet-Jepsen, A., and Bhowmik, A. (2017). Intel realsense stereoscopic depth cameras. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–10.

Kingma, D. and Ba, J. (2014). Adam: A method for stochastic optimization. *International Conference on Learning Representations*.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al. (2017). Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526.

Klambauer, G., Unterthiner, T., Mayr, A., and Hochreiter, S. (2017). Self-normalizing neural networks. In *Advances in neural information processing systems*, pages 971–980.

Kleitsiotis, I., Dimitriou, N., Votis, K., and Tzovaras, D. (2019). Color-guided adaptive support weights for active stereo systems. In *International Conference on Computer Vision Systems*, pages 501–510. Springer.

Koestler, L., Yang, N., Zeller, N., and Cremers, D. (2022). Tandem: Tracking and dense mapping in real-time using deep multi-view stereo. In *Conference on Robot Learning*, pages 34–45. PMLR.

Kolmogorov, V. and Zabih, R. (2002). Multi-camera scene reconstruction via graph cuts. In *European conference on computer vision*, pages 82–96. Springer.

Kosiorek, A. R., Strathmann, H., Zoran, D., Moreno, P., Schneider, R., Mokrá, S., and Rezende, D. J. (2021). Nerf-vae: A geometry aware 3d scene generative model. In *International Conference on Machine Learning*, pages 5742–5752. PMLR.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25.

Kruzhkov, E., Savinykh, A., Karpyshev, P., Kurenkov, M., Yudin, E., Potapov, A., and Tsetserukou, D. (2022). Meslam: Memory efficient slam based on neural fields. In *2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 430–435. IEEE.

Kuznietsov, Y., Stuckler, J., and Leibe, B. (2017). Semi-supervised deep learning for monocular depth map prediction. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6647–6655.

Leutenegger, S., Lynen, S., Bosse, M., Siegwart, R., and Furgale, P. (2015). Keyframe-based visual–inertial odometry using nonlinear optimization. *The International Journal of Robotics Research*, 34(3):314–334.

Levoy, M. and Hanrahan, P. (1996). Light field rendering. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 31–42.

Li, J., Feng, Z., She, Q., Ding, H., Wang, C., and Lee, G. H. (2021a). Mine: Towards continuous depth mpi with nerf for novel view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12578–12588.

Li, K., Niemeyer, M., Navab, N., and Tombari, F. (2023a). Dns slam: Dense neural semantic-informed slam. *arXiv preprint arXiv:2312.00204*.

Li, K., Tang, Y., Prisacariu, V. A., and Torr, P. H. (2022a). Bnv-fusion: Dense 3d reconstruction using bi-level neural volume fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6166–6175.

Li, M., He, J., Jiang, G., and Wang, H. (2024). Ddn-slam: Real-time dense dynamic neural implicit slam with joint semantic encoding. *arXiv preprint arXiv:2401.01545*.

Li, R., Wang, S., and Gu, D. (2020). Deepslam: A robust monocular slam system with unsupervised deep learning. *IEEE Transactions on Industrial Electronics*, 68(4):3577–3587.

Li, R., Wang, S., Long, Z., and Gu, D. (2018a). Undeepvo: Monocular visual odometry through unsupervised deep learning. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 7286–7291. IEEE.

Li, Y., Bu, R., Sun, M., Wu, W., Di, X., and Chen, B. (2018b). Pointcnn: Convolution on x-transformed points. In *Advances in neural information processing systems*, pages 820–830.

Li, Y., Liu, X., Dong, W., Zhou, H., Bao, H., Zhang, G., Zhang, Y., and Cui, Z. (2022b). Deltar: Depth estimation from a light-weight tof sensor and rgb image. In *European Conference on Computer Vision*, pages 619–636. Springer.

Li, Y., Peng, J., Zhang, Y., and Xiong, Z. (2023b). Self-distilled depth from single-shot structured light with intensity reconstruction. *IEEE Transactions on Computational Imaging*.

Li, Z., Niklaus, S., Snavely, N., and Wang, O. (2021b). 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.

Lin, C.-H., Ma, W.-C., Torralba, A., and Lucey, S. (2021). Barf: Bundle-adjusting neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5741–5751.

Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., and Belongie, S. (2017). Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125.

Liu, M., Shi, R., Chen, L., Zhang, Z., Xu, C., Wei, X., Chen, H., Zeng, C., Gu, J., and Su, H. (2023). One-2-3-45++: Fast single image to 3d objects with consistent multi-view generation and 3d diffusion. *arXiv preprint arXiv:2311.07885*.

Liu, M., Xu, C., Jin, H., Chen, L., Varma T, M., Xu, Z., and Su, H. (2024). One-2-3-45: Any single image to 3d mesh in 45 seconds without per-shape optimization. *Advances in Neural Information Processing Systems*, 36.

Liu, S. and Zhu, J. (2023). Efficient map fusion for multiple implicit slam agents. *IEEE Transactions on Intelligent Vehicles*.

Liu, Y., Peng, S., Liu, L., Wang, Q., Wang, P., Theobalt, C., Zhou, X., and Wang, W. (2022). Neural rays for occlusion-aware image-based rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7824–7833.

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

Loshchilov, I. and Hutter, F. (2017). SGDR: stochastic gradient descent with warm restarts. In *International Conference on Learning Representations (ICLR)*.

Luiten, J., Kopanas, G., Leibe, B., and Ramanan, D. (2023). Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. *arXiv preprint arXiv:2308.09713*.

Luo, X., Huang, J.-B., Szeliski, R., Matzen, K., and Kopf, J. (2020). Consistent video depth estimation. *ACM Trans. Graph.*, 39(4).

Mao, Y., Yu, X., Wang, K., Wang, Y., Xiong, R., and Liao, Y. (2023). Ngel-slam: Neural implicit representation-based global consistent low-latency slam system. *arXiv preprint arXiv:2311.09525*.

Martin-Brualla, R., Radwan, N., Sajjadi, M. S., Barron, J. T., Dosovitskiy, A., and Duckworth, D. (2021). 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.

Martinez, M. and Stiefelhagen, R. (2013). Kinect unleashed: Getting control over high resolution depth maps. In *MVA*, pages 247–250.

Matsuki, H., Murai, R., Kelly, P. H., and Davison, A. J. (2023). Gaussian splatting slam. *arXiv preprint arXiv:2312.06741*.

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

Mayer, N., Ilg, E., Hausser, P., Fischer, P., Cremers, D., Dosovitskiy, A., and Brox, T. (2016). A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4040–4048.

McCormac, J., Handa, A., Davison, A., and Leutenegger, S. (2017). Semanticfusion: Dense 3d semantic mapping with convolutional neural networks. In *2017 IEEE International Conference on Robotics and automation (ICRA)*, pages 4628–4635. IEEE.

Meister, S., Hur, J., and Roth, S. (2018). UnFlow: Unsupervised learning of optical flow with a bidirectional census loss. In *AAAI*, New Orleans, Louisiana.

Meshry, M., Goldman, D. B., Khamis, S., Hoppe, H., Pandey, R., Snavely, N., and Martin-Brualla, R. (2019). Neural rerendering in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6878–6887.

Mikhailov, A. (2019). Turbo, an improved rainbow colormap for visualization. *Google AI Blog*.

Mildenhall, B., Hedman, P., Martin-Brualla, R., Srinivasan, P. P., and Barron, J. T. (2022). Nerf in the dark: High dynamic range view synthesis from noisy raw images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16190–16199.

Mildenhall, B., Srinivasan, P. P., Ortiz-Cayon, R., Kalantari, N. K., Ramamoorthi, R., Ng, R., and Kar, A. (2019). Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (TOG)*.

Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., and Ng, R. (2020). Nerf: Representing scenes as neural radiance fields for view synthesis. In *European conference on computer vision*, pages 405–421. Springer.

Ming, Y., Ye, W., and Calway, A. (2022). idf-slam: End-to-end rgb-d slam with neural implicit mapping and deep feature tracking. *arXiv preprint arXiv:2209.07919*.

Mokssit, S., Licea, D. B., Guermah, B., and Ghogho, M. (2023). Deep learning techniques for visual slam: A survey. *IEEE Access*, 11:20026–20050.

Mourikis, A. I. and Roumeliotis, S. I. (2007). A multi-state constraint kalman filter for vision-aided inertial navigation. In *Proceedings 2007 IEEE international conference on robotics and automation*, pages 3565–3572. IEEE.

Müller, T., Evans, A., Schied, C., and Keller, A. (2022). Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)*, 41(4):1–15.

Mur-Artal, R. and Tardós, J. D. (2017a). Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE transactions on robotics*, 33(5):1255–1262.

Mur-Artal, R. and Tardós, J. D. (2017b). Visual-inertial monocular slam with map reuse. *IEEE Robotics and Automation Letters*, 2(2):796–803.

Murez, Z., As, T. v., Bartolozzi, J., Sinha, A., Badrinarayanan, V., and Rabinovich, A. (2020). Atlas: End-to-end 3d scene reconstruction from posed images. In *European conference on computer vision*, pages 414–431. Springer.

Newcombe, R. A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A. J., Kohi, P., Shotton, J., Hodges, S., and Fitzgibbon, A. (2011a). Kinectfusion: Real-time dense surface mapping and tracking. In *2011 10th IEEE international symposium on mixed and augmented reality*, pages 127–136. Ieee.

Newcombe, R. A., Lovegrove, S. J., and Davison, A. J. (2011b). Dtam: Dense tracking and mapping in real-time. In *2011 international conference on computer vision*, pages 2320–2327. IEEE.

Nguyen, P., Karnewar, A., Huynh, L., Rahtu, E., Matas, J., and Heikkila, J. (2021). Rgbd-net: Predicting color and depth images for novel views synthesis. In *2021 International Conference on 3D Vision (3DV)*, pages 1095–1105. IEEE.

Oechsle, M., Peng, S., and Geiger, A. (2021). Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5589–5599.

Or-El, R., Luo, X., Shan, M., Shechtman, E., Park, J. J., and Kemelmacher-Shlizerman, I. (2022). Stylesdf: High-resolution 3d-consistent image and geometry generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13503–13513.

Ortiz, J., Clegg, A., Dong, J., Sucar, E., Novotny, D., Zollhoefer, M., and Mukadam, M. (2022). isdf: Real-time neural signed distance fields for robot perception. In *Robotics: Science and Systems*.

Park, J. J., Florence, P., Straub, J., Newcombe, R., and Lovegrove, S. (2019). Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 165–174.

Park, K., Sinha, U., Barron, J. T., Bouaziz, S., Goldman, D. B., Seitz, S. M., and Martin-Brualla, R. (2021a). Nerfies: Deformable neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5865–5874.

Park, K., Sinha, U., Hedman, P., Barron, J. T., Bouaziz, S., Goldman, D. B., Martin-Brualla, R., and Seitz, S. M. (2021b). Hypernerf: a higher-dimensional representation for topologically varying neural radiance fields. *ACM Transactions on Graphics (TOG)*, 40(6):1–12.

Peng, S., Zhang, Y., Xu, Y., Wang, Q., Shuai, Q., Bao, H., and Zhou, X. (2021). Neural body: Implicit neural representations with structured latent codes for novel view synthesis of dynamic humans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9054–9063.

Penner, E. and Zhang, L. (2017). Soft 3d reconstruction for view synthesis. *ACM Transactions on Graphics (TOG)*, 36(6):1–11.

Piccinelli, L., Sakaridis, C., and Yu, F. (2023). idisc: Internal discretization for monocular depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21477–21487.

Pillai, S., Ambruş, R., and Gaidon, A. (2019). Superdepth: Self-supervised, super-resolved monocular depth estimation. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 9250–9256. IEEE.

Pumarola, A., Corona, E., Pons-Moll, G., and Moreno-Noguer, F. (2021). D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10318–10327.

Qi, C. R., Liu, W., Wu, C., Su, H., and Guibas, L. J. (2018). Frustum pointnets for 3d object detection from rgb-d data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 918–927.

Qi, C. R., Su, H., Mo, K., and Guibas, L. J. (2017a). Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660.

Qi, C. R., Yi, L., Su, H., and Guibas, L. J. (2017b). Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30.

Qiao, R., Kawasaki, H., and Zha, H. (2022). Tide: Temporally incremental disparity estimation via pattern flow in structured light system. *IEEE Robotics and Automation Letters*, 7(2):5111–5118.

Qiao, R., Kawasaki, H., and Zha, H. (2023). Online adaptive disparity estimation for dynamic scenes in structured light systems. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 11186–11193. IEEE.

Qin, T., Li, P., and Shen, S. (2018). Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics*, 34(4):1004–1020.

Qu, D., Yan, C., Wang, D., Yin, J., Xu, D., Zhao, B., and Li, X. (2023). Implicit event-rgbd neural slam. *arXiv preprint arXiv:2311.11013*.

Ranftl, R., Lasinger, K., Hafner, D., Schindler, K., and Koltun, V. (2020). Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE transactions on pattern analysis and machine intelligence*, 44(3):1623–1637.

Ranjan, A., Jampani, V., Balles, L., Kim, K., Sun, D., Wulff, J., and Black, M. J. (2019). Competitive collaboration: Joint unsupervised learning of depth, camera motion, optical flow and motion segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 12240–12249.

Reizenstein, J., Shapovalov, R., Henzler, P., Sbordone, L., Labatut, P., and Novotny, D. (2021). Common objects in 3d: Large-scale learning and evaluation of real-life 3d category reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10901–10911.

Riegler, G., Liao, Y., Donne, S., Koltun, V., and Geiger, A. (2019). Connecting the dots: Learning representations for active monocular depth estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7624–7633.

Rockwell, C., Fouhey, D. F., and Johnson, J. (2021). Pixelsynth: Generating a 3d-consistent experience from a single image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14104–14113.

Rosinol, A., Leonard, J. J., and Carlone, L. (2023). Nerf-slam: Real-time dense monocular slam with neural radiance fields. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3437–3444. IEEE.

Rosu, R. A. and Behnke, S. (2022). Neuralmvs: Bridging multi-view stereo and novel view synthesis. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7. IEEE.

Rusinkiewicz, S. and Levoy, M. (2001). Efficient variants of the icp algorithm. In *Proceedings third international conference on 3-D digital imaging and modeling*, pages 145–152. IEEE.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252.

Ryan Fanello, S., Rhemann, C., Tankovich, V., Kowdle, A., Orts Escolano, S., Kim, D., and Izadi, S. (2016). Hyperdepth: Learning depth from structured light without matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5441–5450.

Ryan Fanello, S., Valentin, J., Kowdle, A., Rhemann, C., Tankovich, V., Ciliberto, C., Davidson, P., and Izadi, S. (2017). Low compute and fully parallel computer vision with hashmatch. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3874–3883.

Sandström, E., Li, Y., Van Gool, L., and Oswald, M. R. (2023). Point-slam: Dense neural point cloud-based slam. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 18433–18444.

Scharstein, D. and Szeliski, R. (1920). A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International journal of computer vision*, 47(1-3):7–42.

Schonberger, J. L. and Frahm, J.-M. (2016). Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113.

Schönberger, J. L., Zheng, E., Frahm, J.-M., and Pollefeys, M. (2016). Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision*, pages 501–518. Springer.

Schops, T., Sattler, T., and Pollefeys, M. (2019). Bad slam: Bundle adjusted direct rgb-d slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 134–144.

Schreiberhuber, S., Weibel, J.-B., Patten, T., and Vincze, M. (2022). Gigadepth: Learning depth from structured light with branching neural networks. In *European Conference on Computer Vision*, pages 214–229. Springer.

Schwarz, K., Liao, Y., Niemeyer, M., and Geiger, A. (2020). Graf: Generative radiance fields for 3d-aware image synthesis. *Advances in Neural Information Processing Systems*, 33.

Shao, S., Pei, Z., Chen, W., Wu, X., and Li, Z. (2023). Nddepth: Normal-distance assisted monocular depth estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7931–7940.

Shoemake, K. (1985). Animating rotation with quaternion curves. In *Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, pages 245–254.

Shreiner, D., Sellers, G., Kessenich, J., and Licea-Kane, B. (2013). *OpenGL programming guide: The Official guide to learning OpenGL, version 4.3*. Addison-Wesley.

Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*.

Srinivasan, P. P., Deng, B., Zhang, X., Tancik, M., Mildenhall, B., and Barron, J. T. (2021). Nerv: Neural reflectance and visibility fields for relighting and view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7495–7504.

Srinivasan, P. P., Tucker, R., Barron, J. T., Ramamoorthi, R., Ng, R., and Snavely, N. (2019). Pushing the boundaries of view extrapolation with multiplane images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 175–184.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Straub, J., Whelan, T., Ma, L., Chen, Y., Wijmans, E., Green, S., Engel, J. J., Mur-Artal, R., Ren, C., Verma, S., Clarkson, A., Yan, M., Budge, B., Yan, Y., Pan, X., Yon, J., Zou, Y., Leon, K., Carter, N., Briales, J., Gillingham, T., Mueggler, E., Pesqueira, L., Savva, M., Batra, D., Strasdat, H. M., Nardi, R. D., Goesele, M., Lovegrove, S., and Newcombe, R. (2019). The Replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*.

Sturm, J., Engelhard, N., Endres, F., Burgard, W., and Cremers, D. (2012). A benchmark for the evaluation of rgb-d slam systems. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 573–580. IEEE.

Sucar, E., Liu, S., Ortiz, J., and Davison, A. J. (2021). imap: Implicit mapping and positioning in real-time. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6229–6238.

Sucar, E., Wada, K., and Davison, A. (2020). Nodeslam: Neural object descriptors for multi-view shape reconstruction. In *2020 International Conference on 3D Vision (3DV)*, pages 949–958. IEEE.

Suganyadevi, S., Seethalakshmi, V., and Balasamy, K. (2022). A review on deep learning in medical image analysis. *International Journal of Multimedia Information Retrieval*, 11(1):19–38.

Suhail, M., Esteves, C., Sigal, L., and Makadia, A. (2022). Generalizable patch-based neural rendering. In *European Conference on Computer Vision*, pages 156–174. Springer.

Sun, C., Sun, M., and Chen, H.-T. (2022a). Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5459–5469.

Sun, J., Chen, X., Wang, Q., Li, Z., Averbuch-Elor, H., Zhou, X., and Snavely, N. (2022b). Neural 3d reconstruction in the wild. In *ACM SIGGRAPH 2022 Conference Proceedings*, pages 1–9.

Sun, J., Xie, Y., Chen, L., Zhou, X., and Bao, H. (2021). Neuralrecon: Real-time coherent 3d reconstruction from monocular video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15598–15607.

Sünderhauf, N., Pham, T. T., Latif, Y., Milford, M., and Reid, I. (2017). Meaningful maps with object-oriented semantic mapping. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5079–5085. IEEE.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9.

Tang, C. and Tan, P. (2018). Ba-net: Dense bundle adjustment networks. In *International Conference on Learning Representations*.

Tang, J., Ren, J., Zhou, H., Liu, Z., and Zeng, G. (2023a). Dreamgaussian: Generative gaussian splatting for efficient 3d content creation. *arXiv preprint arXiv:2309.16653*.

Tang, Y., Zhang, J., Yu, Z., Wang, H., and Xu, K. (2023b). Mips-fusion: Multi-implicit-submaps for scalable and robust online neural rgb-d reconstruction. *ACM Transactions on Graphics (TOG)*, 42(6):1–16.

Tateno, K., Tombari, F., Laina, I., and Navab, N. (2017). Cnn-slam: Real-time dense monocular slam with learned depth prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Teed, Z. and Deng, J. (2019). Deepv2d: Video to depth with differentiable structure from motion. In *International Conference on Learning Representations*.

Teed, Z. and Deng, J. (2021). Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras. *Advances in Neural Information Processing Systems*, 34:16558–16569.

Thomas, H., Qi, C. R., Deschaud, J.-E., Marcotegui, B., Goulette, F., and Guibas, L. J. (2019). Kpconv: Flexible and deformable convolution for point clouds. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6411–6420.

Toshev, A. and Szegedy, C. (2014). Deeppose: Human pose estimation via deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1653–1660.

Trevithick, A. and Yang, B. (2021). Grf: Learning a general radiance field for 3d representation and rendering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15182–15192.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Verbin, D., Hedman, P., Mildenhall, B., Zickler, T., Barron, J. T., and Srinivasan, P. P. (2022). Ref-nerf: Structured view-dependent appearance for neural radiance fields. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5481–5490. IEEE.

Von Stumberg, L., Usenko, V., and Cremers, D. (2018). Direct sparse visual-inertial odometry using dynamic marginalization. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2510–2517. IEEE.

Wang, J., Bleja, T., and Agapito, L. (2022a). Go-surf: Neural feature grid optimization for fast, high-fidelity rgb-d surface reconstruction. In *2022 International Conference on 3D Vision (3DV)*, pages 433–442. IEEE.

Wang, J., Wang, P., Long, X., Theobalt, C., Komura, T., Liu, L., and Wang, W. (2022b). Neuris: Neural reconstruction of indoor scenes using normal priors. In *European Conference on Computer Vision*, pages 139–155. Springer.

Wang, N., Zhang, Y., Li, Z., Fu, Y., Liu, W., and Jiang, Y.-G. (2018a). Pixel2mesh: Generating 3d mesh models from single rgb images. In *Proceedings of the European conference on computer vision (ECCV)*, pages 52–67.

Wang, P., Liu, L., Liu, Y., Theobalt, C., Komura, T., and Wang, W. (2021a). Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *NeurIPS*.

Wang, Q., Wang, Z., Genova, K., Srinivasan, P. P., Zhou, H., Barron, J. T., Martin-Brualla, R., Snavely, N., and Funkhouser, T. (2021b). Ibrnet: Learning multi-view image-based rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4690–4699.

Wang, S., Suo, S., Ma, W.-C., Pokrovsky, A., and Urtasun, R. (2018b). Deep parametric continuous convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2589–2597.

Wang, Y., Wang, P., Yang, Z., Luo, C., Yang, Y., and Xu, W. (2019). Unos: Unified unsupervised optical-flow and stereo-depth estimation by watching videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8071–8081.

Wang, Z., Bovik, A. C., Sheikh, H. R., and Simoncelli, E. P. (2004). Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612.

Wang, Z., Li, L., Shen, Z., Shen, L., and Bo, L. (2022c). 4k-nerf: High fidelity neural radiance fields at ultra high resolutions. *arXiv preprint arXiv:2212.04701*.

Wang, Z., Wu, S., Xie, W., Chen, M., and Prisacariu, V. A. (2021c). Nerf–: Neural radiance fields without known camera parameters. *arXiv preprint arXiv:2102.07064*.

Weder, S., Schonberger, J. L., Pollefeys, M., and Oswald, M. R. (2021). Neuralfusion: Online depth fusion in latent space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3162–3172.

Wei, X., Zhang, Y., Li, Z., Fu, Y., and Xue, X. (2020). Deepsfm: Structure from motion via deep bundle adjustment. In *European conference on computer vision*, pages 230–247. Springer.

Wei, Y., Liu, S., Rao, Y., Zhao, W., Lu, J., and Zhou, J. (2021). Nerfingmvs: Guided optimization of neural radiance fields for indoor multi-view stereo. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5610–5619.

Whelan, T., Kaess, M., Fallon, M., Johannsson, H., Leonard, J., and McDonald, J. (2012). Kintinuous: Spatially extended kinectfusion. In *RSS Workshop on RGB-D: Advanced Reasoning with Depth Cameras*.

Whelan, T., Leutenegger, S., Salas-Moreno, R., Glocker, B., and Davison, A. (2015). Elasticfusion: Dense slam without a pose graph. In *Robotics: Science and Systems (RSS)*. Robotics: Science and Systems.

Wu, W., Qi, Z., and Fuxin, L. (2019). Pointconv: Deep convolutional networks on 3d point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9621–9630.

Wu, X., Xu, J., Zhu, Z., Bao, H., Huang, Q., Tompkin, J., and Xu, W. (2022). Scalable neural indoor scene rendering. *ACM Transactions on Graphics (TOG)*, 41(4):1–16.

Wu, Y. and He, K. (2018). Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19.

Xia, Y., Tang, H., Timofte, R., and Van Gool, L. (2022). Sinerf: Sinusoidal neural radiance fields for joint pose estimation and scene reconstruction. *arXiv preprint arXiv:2210.04553*.

Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J. M., and Luo, P. (2021). Segformer: Simple and efficient design for semantic segmentation with transformers. *Advances in Neural Information Processing Systems*, 34:12077–12090.

Xie, H., Yao, H., Sun, X., Zhou, S., and Zhang, S. (2019). Pix2vox: Context-aware 3d reconstruction from single and multi-view images. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2690–2698.

Xie, J., Girshick, R., and Farhadi, A. (2016). Deep3d: Fully automatic 2d-to-3d video conversion with deep convolutional neural networks. In *European Conference on Computer Vision*, pages 842–857. Springer.

Xu, J., Peng, L., Cheng, H., Li, H., Qian, W., Li, K., Wang, W., and Cai, D. (2023a). Mononerd: Nerf-like representations for monocular 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6814–6824.

Xu, M., Zhan, F., Zhang, J., Yu, Y., Zhang, X., Theobalt, C., Shao, L., and Lu, S. (2023b). Wavenerf: Wavelet-based generalizable neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 18195–18204.

Xu, Q. and Tao, W. (2020). Learning inverse depth regression for multi-view stereo with correlation cost volume. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34 (07), pages 12508–12515.

Xu, Y., Fan, T., Xu, M., Zeng, L., and Qiao, Y. (2018). Spidercnn: Deep learning on point sets with parameterized convolutional filters. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 87–102.

Xu, Z., Niu, J., Li, Q., Ren, T., and Chen, C. (2024). Nid-slam: Neural implicit representation-based rgb-d slam in dynamic environments. *arXiv preprint arXiv:2401.01189*.

Yan, C., Qu, D., Wang, D., Xu, D., Wang, Z., Zhao, B., and Li, X. (2023). Gs-slam: Dense visual slam with 3d gaussian splatting. *arXiv preprint arXiv:2311.11700*.

Yan, Z., Tian, Y., Shi, X., Guo, P., Wang, P., and Zha, H. (2021). Continual neural mapping: Learning an implicit scene representation from sequential observations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15782–15792.

Yang, H., Hong, L., Li, A., Hu, T., Li, Z., Lee, G. H., and Wang, L. (2023a). Contranerf: Generalizable neural radiance fields for synthetic-to-real novel view synthesis via contrastive learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16508–16517.

Yang, J., Mao, W., Alvarez, J. M., and Liu, M. (2020). Cost volume pyramid based depth inference for multi-view stereo. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4877–4886.

Yang, X., Li, H., Zhai, H., Ming, Y., Liu, Y., and Zhang, G. (2022a). Vox-fusion: Dense tracking and mapping with voxel-based neural implicit representation. In *2022 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 499–507. IEEE.

Yang, X., Lin, G., and Zhou, L. (2023b). Single-view 3d mesh reconstruction for seen and unseen categories. *IEEE Transactions on Image Processing*.

Yang, X., Ma, Z., Ji, Z., and Ren, Z. (2023c). Gedepth: Ground embedding for monocular depth estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12719–12727.

Yang, X., Ming, Y., Cui, Z., and Calway, A. (2022b). Fd-slam: 3-d reconstruction using features and dense matching. In *2022 International Conference on Robotics and Automation (ICRA)*, page 8040–8046.

Yang, Z., Gao, X., Zhou, W., Jiao, S., Zhang, Y., and Jin, X. (2023d). Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. *arXiv preprint arXiv:2309.13101*.

Yao, Y., Luo, Z., Li, S., Fang, T., and Quan, L. (2018). Mvsnet: Depth inference for unstructured multi-view stereo. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 767–783.

Yao, Y., Luo, Z., Li, S., Shen, T., Fang, T., and Quan, L. (2019). Recurrent mvsnet for high-resolution multi-view stereo depth inference. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5525–5534.

Yariv, L., Gu, J., Kasten, Y., and Lipman, Y. (2021). Volume rendering of neural implicit surfaces. *Advances in Neural Information Processing Systems*, 34.

Yariv, L., Kasten, Y., Moran, D., Galun, M., Atzmon, M., Ronen, B., and Lipman, Y. (2020). Multiview neural surface reconstruction by disentangling geometry and appearance. *Advances in Neural Information Processing Systems*, 33:2492–2502.

Yen-Chen, L., Florence, P., Barron, J. T., Rodriguez, A., Isola, P., and Lin, T.-Y. (2021). inerf: Inverting neural radiance fields for pose estimation. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1323–1330. IEEE.

Yin, Z. and Shi, J. (2018). Geonet: Unsupervised learning of dense depth, optical flow and camera pose. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1983–1992.

Yu, A., Ye, V., Tancik, M., and Kanazawa, A. (2021). 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.

Yu, Z., Chen, A., Huang, B., Sattler, T., and Geiger, A. (2023). Mip-splatting: Alias-free 3d gaussian splatting. *arXiv preprint arXiv:2311.16493*.

Yugay, V., Li, Y., Gevers, T., and Oswald, M. R. (2023). Gaussian-slam: Photo-realistic dense slam with gaussian splatting. *arXiv preprint arXiv:2312.10070*.

Zhan, H., Garg, R., Saroj Weerasekera, C., Li, K., Agarwal, H., and Reid, I. (2018). Unsupervised learning of monocular depth estimation and visual odometry with deep feature reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 340–349.

Zhang, J., Ren, D., Cai, Z., Yeo, C. K., Dai, B., and Loy, C. C. (2022). Monocular 3d object reconstruction with gan inversion. In *European Conference on Computer Vision*, pages 673–689. Springer.

Zhang, J., Yang, G., Tulsiani, S., and Ramanan, D. (2021). Ners: Neural reflectance surfaces for sparse-view 3d reconstruction in the wild. *Advances in Neural Information Processing Systems*, 34:29835–29847.

Zhang, R., Isola, P., Efros, A. A., Shechtman, E., and Wang, O. (2018a). The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595.

Zhang, W., Sun, T., Wang, S., Cheng, Q., and Haala, N. (2023a). Hi-slam: Monocular real-time dense mapping with hybrid implicit fields. *IEEE Robotics and Automation Letters*.

Zhang, X., Chen, Z., Wei, F., and Tu, Z. (2023b). Uni-3d: A universal model for panoptic 3d scene reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9256–9266.

Zhang, Y., Ji, P., Wang, A., Mei, J., Kortylewski, A., and Yuille, A. (2023c). 3d-aware neural body fitting for occlusion robust 3d human pose estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9399–9410.

Zhang, Y., Khamis, S., Rhemann, C., Valentin, J., Kowdle, A., Tankovich, V., Schoenberg, M., Izadi, S., Funkhouser, T., and Fanello, S. (2018b). Activestereonet: End-to-end self-supervised learning for active stereo systems. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 784–801.

Zhao, C., Poggi, M., Tosi, F., Zhou, L., Sun, Q., Tang, Y., and Mattoccia, S. (2023). Gasmono: Geometry-aided self-supervised monocular depth estimation for indoor scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16209–16220.

Zhi, S., Bloesch, M., Leutenegger, S., and Davison, A. J. (2019). Scenecode: Monocular dense semantic reconstruction using learned encoded scene representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11776–11785.

Zhou, J., Chen, K., Xu, L., Dou, Q., and Qin, J. (2023). Deep fusion transformer network with weighted vector-wise keypoints voting for robust 6d object pose estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13967–13977.

Zhou, T., Brown, M., Snavely, N., and Lowe, D. G. (2017). Unsupervised learning of depth and ego-motion from video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1851–1858.

Zhou, T., Tucker, R., Flynn, J., Fyffe, G., and Snavely, N. (2018). Stereo magnification: learning view synthesis using multiplane images. *ACM Transactions on Graphics (TOG)*, 37(4):1–12.

Zhou, Z. and Dong, Q. (2023). Two-in-one depth: Bridging the gap between monocular and binocular self-supervised depth estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9411–9421.

Zhu, S., Wang, G., Blum, H., Liu, J., Song, L., Pollefeys, M., and Wang, H. (2023a). Sni-slam: Semantic neural implicit slam. *arXiv preprint arXiv:2311.11016*.

Zhu, Z., Peng, S., Larsson, V., Cui, Z., Oswald, M. R., Geiger, A., and Pollefeys, M. (2023b). Nicer-slam: Neural implicit scene encoding for rgb slam. *arXiv preprint arXiv:2302.03594*.

Zhu, Z., Peng, S., Larsson, V., Xu, W., Bao, H., Cui, Z., Oswald, M. R., and Pollefeys, M. (2022). Nice-slam: Neural implicit scalable encoding for slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12786–12796.

Zou, Y., Luo, Z., and Huang, J.-B. (2018). Df-net: Unsupervised joint learning of depth and flow using cross-task consistency. In *Proceedings of the European conference on computer vision (ECCV)*, pages 36–53.

Zou, Z.-X., Huang, S.-S., Cao, Y.-P., Mu, T.-J., Shan, Y., and Fu, H. (2022). Mononeuralfusion: Online monocular neural 3d reconstruction with geometric priors. *arXiv preprint arXiv:2209.15153*.

Mohammad Mahdi Johari

+41 76-7211371
Lausanne, Switzerland
mohammad.johari@epfl.ch
mohammadjohari.github.io

Machine Learning Researcher

Education

P.h.D in Machine Learning, *Swiss Federal Institute of Technology Lausanne (EPFL)*	Feb. 2020 — June 2024
M. Sc. in Electrical Engineering, *Sharif University of Technology*	Sept. 2014 — Sept. 2016
B. Sc. in Electrical Engineering, *University of Tehran*	Sept. 2010 — Sept. 2014

Skills

Technical	Machine Learning, Deep Learning, Computer Vision, 3D Computer Vision, Computer Graphics, 3D Representation and Reconstruction, 3D Depth Estimation, Novel View Synthesis, Neural Radiance Fields, SLAM, Graph Analysis, Geometric Deep Learning, Python, PyTorch, PyTorch Geometric, TensorFlow, Git, LaTeX, MATLAB, VHDL, Verilog, C++, C#, HTML, Linux Shell, Java
Personal	Problem Solving, Multitasking, Adaptability, Self-motivation, Time management, Communication, Research skills, Intercultural collaboration, Creativity, Attention to details, Decision making, Project management

Technical Experience

Efficient 3D Face Representation for Real-Time Rendering (Research Internship) **April 2023 — Nov. 2023**

Apple *Zurich, Switzerland*

- This research introduces an occlusion-aware generalizable human face representation capable of view-dependent rendering. Exploiting both explicit and implicit 3D priors in a hybrid manner, the method generates high-quality 3D-consistent novel views while it is efficient enough to run in real-time on a high-end device.

ESLAM: Efficient Dense SLAM Based on Hybrid Representation of Signed Distance Fields (CVPR 2023) **April 2022 — March. 2023**

Swiss Federal Institute of Technology Lausanne (EPFL), Idiap Research Institute *Lausanne, Switzerland*

- Project page: https://www.idiap.ch/paper/eslam
- This research presents an efficient implicit neural representation method for Simultaneous Localization and Mapping (SLAM). ESLAM reads RGB-D frames with unknown camera poses in a sequential manner and incrementally reconstructs the scene representation while estimating the current camera position in the scene. We incorporate the latest advances in Neural Radiance Fields (NeRF) into a SLAM system, resulting in an efficient and accurate dense visual SLAM method.

GeoNeRF: Generalizing Neural Radiance Fileds for Novel View Synthesis (CVPR 2022) **June 2021 — Feb. 2022**

Swiss Federal Institute of Technology Lausanne (EPFL), Idiap Research Institute *Lausanne, Switzerland*

- Project page: https://www.idiap.ch/paper/geonerf
- This research presents a generalizable photorealistic novel view synthesis method based on neural radiance fields. The approach consists of two main stages: a geometry reasoner and a renderer. To render a novel view, the geometry reasoner first constructs cascaded cost volumes for each nearby source view. Then, using a Transformer-based attention mechanism and the cascaded cost volumes, the renderer infers geometry and appearance and renders detailed images via volume rendering techniques.

DepthInSpace: Monocular 3D Depth Estimation Using Structured-Light Camera (ICCV 2021) **Feb. 2020 — May 2021**

Swiss Federal Institute of Technology Lausanne (EPFL), Idiap Research Institute *Lausanne, Switzerland*

- Project page: https://www.idiap.ch/paper/depthinspace

- This research presents a self-supervised deep-learning method for depth estimation using a structured-light camera. The model first uses estimated optical flow from ambient information of multiple video frames as a complementary guide for training a single-frame depth estimation network. Utilizing optical flow, it also fuses the data of multiple video frames to get a more accurate depth map. Lastly, these more precise fused depth maps are used as self-supervision for fine-tuning a single-frame depth estimation network to improve its performance.

Semantic Segmentation of Aerial Images (Machine Learning Engineer) **June 2019 — Jan. 2020**

Cafe Bazaar *Tehran, Iran*

- As a member of the research team for the Balad application (a customized navigation app for Iran), I built a model upon Pyramid Scene Parsing Network (PSP Net) to carry out building roof segmentation utilizing aerial satellite images. The segmentation is beneficial in the precise locating of paramount buildings in the navigation map.

Automatic Image Colorization Using Artificial Intelligence (ICASSP 2020 and Neurocomputing 2020) **Oct. 2017 — May 2019**

Sharif University of Technology *Tehran, Iran*

- This research investigates the automatic colorization of gray-scale images. To accomplish the goal, a two-stage cycle-consistent architecture based on Generative Adversarial Networks (GAN) is proposed. The work resulted in the publishing of two articles.

Smart Home Package (Co-Founder and Software Engineer) **Nov. 2016 — Sept. 2017**

Griffin Smart Home *Tehran, Iran*

- Designing and developing Android (Java) and IOS (Objective-C) applications to control the smart home package via local WiFi network, SMS, Internet, or a scheduled plan. The package includes Light Controller (Dimmer or Switch), Automatic Door Opener, Surveillance Camera Controller, RGB Light Controller, Thermostatic Controller, Global IR Remote Controller, and Safety Sensor Controller (Gas, Smoke, and Motion Sensors).

M.Sc. Thesis (IRS 2016) **Sept. 2014 — Sept. 2016**

Sharif University of Technology *Tehran, Iran*

- This research aims at designing a platform to classify various airborne objects based on their micro-Doppler effects of the echo signal. Numerous statistical features based on the Recurrence Plot are extracted and fed to a Multi-class Support Vector Machine classifier. The features are claimed to be robust against natural movements, direction, and aspect angle of the objects.

B.Sc. Thesis **Sept. 2013 — Sept. 2014**

University of Tehran *Tehran, Iran*

- This research addresses the problem of locating smartphones in an indoor environment by measuring the received power from available WiFi access points. The problem is solved in a semi-supervised fashion. As such, the data is fit into a multivariate Gaussian mixture model in order to exploit unlabeled data, which is abundantly available after the classifier is learned.

Digital Logic Design Lab (Lab Assistant) **Sept. 2012 — Sept. 2013**

University of Tehran *Tehran, Iran*

- Implementing several digital instruments on FPGA, such as a Digital Function Generator, a Digital Voltmeter, and a Digital Oscilloscope utilizing a VGA monitor.

Publications

- Johari, Mohammad Mahdi, Camilla Carta, and François Fleuret. "ESLAM: Efficient Dense SLAM System Based on Hybrid Representation of Signed Distance Fields." *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*. 2023. URL: https://www.idiap.ch/paper/eslam.
- Johari, Mohammad Mahdi, Yann Lepoittevin, and François Fleuret. "GeoNeRF: Generalizing NeRF with Geometry Priors." *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022. URL: https://www.idiap.ch/paper/geonerf.
- Johari, Mohammad Mahdi, Camilla Carta, and François Fleuret. "DepthInSpace: Exploitation and Fusion of Multiple Video Frames for Structured-Light Depth Estimation." *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021. URL: https://www.idiap.ch/paper/depthinspace.
- Johari, Mohammad Mahdi, and Hamid Behroozi. "Context-aware colorization of gray-scale images utilizing a cycle-consistent generative adversarial network architecture." *Neurocomputing* 407 (2020): 94-104.
- Johari, Mohammad Mahdi, and Hamid Behroozi. "Gray-Scale Image Colorization Using Cycle-Consistent Generative Adversarial Networks with Residual Structure Enhancer." *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020.
- Johari, Mohammad Mahdi, and Mohammad Mahdi Nayebi. "Robust airborne target recognition based on recurrence plot quantification of micro-Doppler radar signatures." *17th International Radar Symposium (IRS)*. IEEE, 2016.

Reviewing Experience

- Reviewing one paper for *ACM Transactions on Graphics*, 2023.
- Reviewing four papers for *International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.
- Reviewing one paper for *Imaging Science Journal*, 2022.
- Reviewing one paper for *IEEE Transactions on Image Processing*, 2021.
- Reviewing one paper for *International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Teaching Experience

Teaching Assistant for *Fundamentals in Statistical Pattern Recognition* (2023)	Instructor: Dr. Sébastien Marcel
Teaching Assistant for *Deep Learning* (2021 and 2022)	Instructor: Prof. François Fleuret
Teaching Assistant for *Probability and Statistics* (2016)	Instructor: Prof. Mohammad Mahdi Nayebi
Teaching Assistant for *Digital Communication Systems* (2014)	Instructor: Prof. Amir Masoud Rabiei
Lab Assistant for *Digital Logic Design Lab* (2013 and 2014)	Instructor: Prof. Zainalabedin Navabi

Awards and Honors

- Ranked **1st** in The 19th National Scientific Olympiad for University Students in Electrical Engineering Field, 2014.
- Ranked **4th** among all the Electrical Engineering students of University of Tehran
- Received a fellowship for Graduate Studies at Sharif University of Technology, in 2014.
- Ranked **200th** among almost 400,000 participants in the Nationwide Iranian Universities Entrance Exam (Konkur) in the field of Mathematics and Physics, 2010.