ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

CVL Computer Vision Lab

Online Sensor-Agnostic Uncertainty Learning for SLAM

Leveraging Implicit Learning of Aleatoric Uncertainty for Neural Implicit Representations

Master's Thesis

Kevin Ta
M. Sc. in Robotics, Systems and Control

Advisors: Erik Sandström, Prof. Dr. Martin R. Oswald
Supervisor: Prof. Dr. Luc van Gool

December 1, 2022

Abstract

Neural implicit scene representations, such as neural radiance fields, have shown to be powerful tools in the field of 3D reconstruction and novel view synthesis. Such representations have the benefit of being fully continuous and to have some degree of extrapolative power. In particular, recent work has shown that such a representation can be used as the sole representation of online SLAM systems. Building on one such work, NICE-SLAM, we augment the reconstruction and SLAM pipelines by learning sensor uncertainty in an online fashion and without supervision from ground truth meshes or depth maps. We show that scaling the objective function by the true error achieves improvements across tracking, reconstruction, and rendering on a synthetic dataset under two different noise model assumptions. We show that a learned uncertainty can improve performance using just 2D feature maps constructed from the depth maps inputs, as well as an approach that leverages the volume rendering and accumulation of uncertainty in a 3D grid of features. We then extend this work to perform two-sensor fusion and show further improvements to depth rendering. This cursory exploration is able to approach or exceed the performance of its constituent single sensor results. This work furthers our understanding of neural implicit representations and how to leverage online learned uncertainty in fusing multiple sensor sources.

Acknowledgements

> I want everyone I've loved and liked to know I've appreciated them very much. [...] I hope my life was worth your time, as yours was worth every second of mine.
>
> *Mike McGee*
> *Before we Leave*

My last name, Ta, is an English romanization of an archaic Vietnamese translation of the Chinese surname 謝, meaning "thanks" or, as I like to think of it, "gratitude" or "gratefulness." I want to take this time to express gratitude to all those who have helped me on my journey throughout my time in Switzerland, throughout my master's, and throughout my life.

To my mom, my dad, and my brother, you have been supporting me and reassuring me as I took the big step of moving halfway across the globe to live and to learn. Thank you for encouraging me to keep going, for reminding me that I always have a place to call home, and providing me, despite all the hardships of our upbringing, a chance to dream of a better life. I hope to live up to your expectation and to return to you kindness and joy, to share in them, and to cherish them, for all the time we have been allotted.

To Dr. Mahsa Khalili, you have been such a patient and encouraging mentor from my time at the CARIS Lab to now. Your approach towards learning and conducting research has been the foundation for how I mould my own approach. Your didactic guidance has informed the way I think about experimentation, how I write, how I try to form my arguments. I'm truly grateful to have collaborated with you across two publications last year. And beyond your academic influence, I am grateful for your friendship, your support, and your honesty.

To Erik Sandström, your guidance across my thesis has made this process so much more productive and collaborative. Your ability to explain concepts and to guide my understanding made me feel supported and well-directed in this research. I am very grateful for your patience and willingness to indulge my my tangents and questions. Thank you again for your understanding and your mentorship over the past seven months.

To Jessica Bo, thank you for being a friend to lean on when I first arrived in Switzerland. For us to start our master's in a global pandemic, your presence eased the transition to a new school, to a new city, to a new country. I'm glad we continued to work together and to publish together, our work from our undergraduate program. Your tireless work ethic and insatiable drive and adventurous spirit continue to inspire me, and

I'm glad we got to study here, together. From our first year in university, to this last year of our master's, I'm truly grateful for your friendship.

To Carter Fang, our last seven months in Switzerland have been a highlight of my life. But beyond the traveling, or the completion of our master's, I am most grateful for your friendship, your kindness, and your inexhaustible thoughtfulness. If there is one thing, one invaluable thing, I have gained in my studies here, it is your friendship. It is a friendship I hope to carry forward for the rest of my life. Thank you for doing the thankless work of reviewing my manuscripts, for being a fixture of our book club every week, and for all the struggling and the overcoming we've had to endure. Thank you for being there, your thoughtfulness, your friendship, and all.

To Kelvin Koon, they say kindred spirits are hard to come by. I often think that our meeting was such a chance encounter, a friendship that's only existed for a year, but feels like longer. It's a lonely thing at times to study in a foreign country, but when you have friends like you, it can be easy to forget. To have someone to whom you can speak genuinely with, to confide in your concerns and your aspirations, in your trials and your triumphs, from the mundane to the extraordinary, is such a privilege. Your empathy, your ambition, and your endless desire for self-improvement are all aspects of you that I genuinely admire. Thank you for your friendship and for being a pillar of support this past year.

There are so many more people who have supported me throughout my studies and I could spend a lifetime expressing my gratitude to each and every one of them. From the longstanding to the momentary paths crossing, your life is marked by each and every relationship, each and every interaction. So thank you, to everyone who has added to the pages of my life. This book is richer for it and for that... what can I say, except that I'm grateful?

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Robots in the real world are often deployed in unexplored environments in which they must *map*—*i.e.* identify their surroundings—and *localize*—*i.e.* determine their location relative to their surroundings. This field of work is described under Simultaneous Localization and Mapping (SLAM). SLAM has been an area of active research within the last 40 years, with focus areas involving exploration into improving accuracy, developing strategies to recover from localization failures, and exploring new scene representations. Over the last decade, the advances in parallel processing and deep learning have dramatically changed how many of the state-of-the-art computer vision tasks are approached. Within 3D vision and SLAM tasks, however, deep learning has mainly been applied to sub-tasks, and is still mostly inferior to classical SLAM pipelines.

One approach for performing SLAM is through the dense-SLAM framework. In this framework, the scene is densely recorded, as opposed to only recording a set of sparse feature-based points. Retrieving such scene reconstructions is thus enabled by dense-SLAM frameworks, generally at the cost of increased memory usage to store scene details—*e.g.* storing occupancy or signed distance functions (SDF) in a voxel grid. In particular, the rise of RGBD sensors has accelerated the development of dense-SLAM methods.

The Microsoft Kinect sensor, shown in Figure 1.1, was first released in 2010 and presented the first widely available RGB-D camera for commercial and research applications. This sensor uses structured-light (SL) to generate dense depth maps. Since then, SL methods have become prevalent. However, newer sensors tend to utilize time-of-flight (ToF) technology—*e.g.* Azure Kinect v2—to achieve greater robustness. In recent years, commercial devices like mobile phones are increasingly equipped with multiple image sensors and even built-in depth sensors.

Figure 1.1: A Microsoft Kinect and Intel RealSense D455 RGBD sensor.

As more and more sensing modalities are developed and becoming integrated into robotics systems, and the advent of faster compute and parallelization improves their practicality, SLAM systems are naturally evolving to exploit these advances. In particular, the rise of integrated sensors in common consumer products provide plentiful, but often noisy, data that can be improved via uncertainty estimation. Constructing scenes using *neural implicit methods* has become of interest due to its ability to generate impressive 3D reconstruction, while ensuring continuous watertight representations. This work looks to build and improve

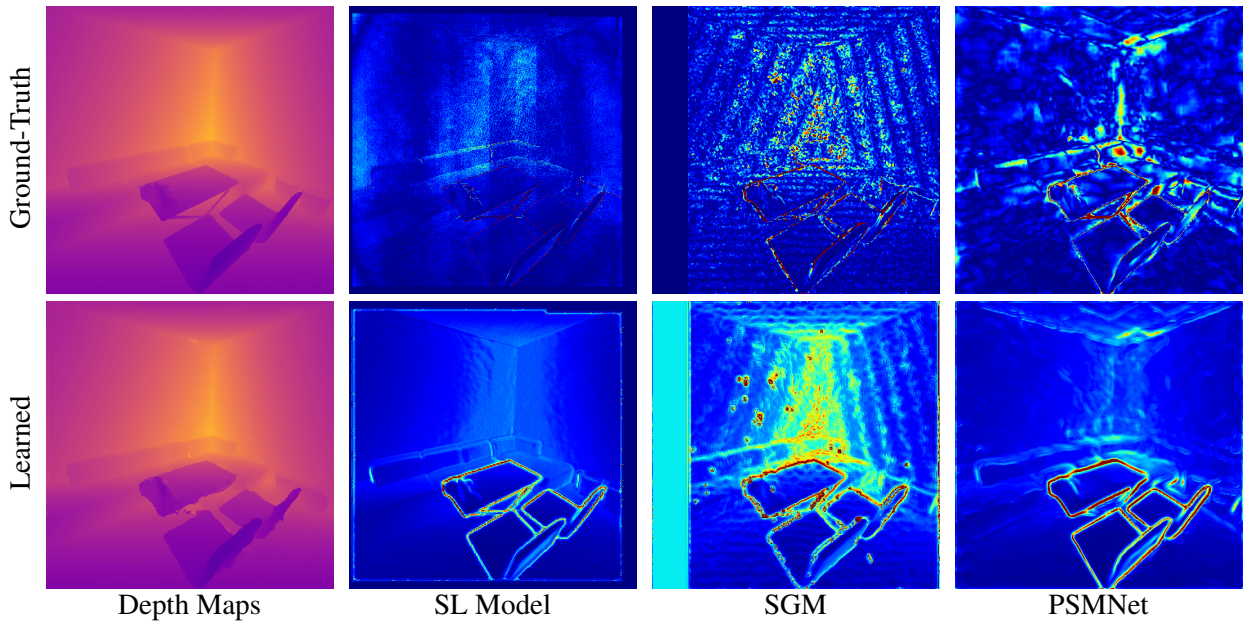


Figure 1.2: (Top) GT depth map and GT errors. (Bottom) Learned depth map and uncertainty.

upon a neural implicit SLAM pipeline, NICE-SLAM [46], using implicit methods for learning uncertainty. In this work, we aim to learn sensor-agnostic uncertainty, shown in Figure 1.2, in an online fashion without supervision from ground-truth meshes or depth maps. Such information can improve reconstruction and rendering, while also elegantly enabling multi-sensor systems.

1.1 Focus of this Work

NICE-SLAM provides a foundational work in investigating neural-implicit scene representations for dense-SLAM systems. It has shown impressive results in scene reconstruction and closes some of the performance gaps compared to classical methods. NICE-SLAM addresses issues with catastrophic forgetting by ensuring local feature updates in the implicit representation and preserving scene geometry from previously viewed regions. This approach, however, has some deficiencies and areas for extension that we aim to address in this work.

NICE-SLAM is dependent on accurate depth ranging for its inputs. We find in this work that the presence of measurement noise has a significant impact on the quality of reconstruction, rendering, and tracking. As all sensors and ranging methods exhibit sensor- or method-dependent noise, this dependency presents a serious challenge in transitioning NICE-SLAM from a synthetic environment to a real-world environment. Leveraging the true error for scaling can further inform 3D reconstruction and approach the performance when using noise-free depth maps.

In practice, ground truth errors are unavailable and we must learn uncertainty implicitly in the training process or operation of such a system. In this work, we aim to learn uncertainty in an online fashion jointly with the scene representation and regardless of the sensor-specific noise distribution. This task is performed without supervision or access to the ground truth information. Such an approach in learning uncertainty has the potential to further close the gap between neural implicit methods and classical methods. Additionally, this work is a key component in unlocking the NICE-SLAM framework for multi-sensor and multi-agent

settings.

This work thus extends the NICE-SLAM framework using implicitly learned uncertainty to guide the single-sensor scene reconstruction, rendering, and tracking. We provide a theoretical foundation for how each sampled ray can be constructed into a probabilistic loss function based on previous works in leveraging uncertainty in computer vision. We showcase results on how our method generally improves on standard performance metrics using a combination of synthetic and real world datasets. Following the single-sensor evaluation, this work explores the fusion of depth maps generated by two aligned and synchronized sensors to achieve performance gains in mesh accuracy and depth rendering. Finally, we discuss the implications of our work and the next steps to incorporate different sensors and to handle multiple agent environments.

1.2 Thesis Organization

This thesis is organized into six chapters that introduce our uncertainty extension to NICE-SLAM, an approach to implicitly learning uncertainty in an online fashion regardless of sensor-specific noise distributions. In Chapter 2, we provide a brief introduction to visual SLAM and dense SLAM. The chapter goes on to discuss the recent trends in deep learning for 3D scene reconstruction and novel view synthesis using neural implicit methods, before exploring their applications in SLAM systems. We end the chapter with a discussion of uncertainty in deep learning. Chapter 3 goes on to discuss the theoretical motivations in constructing our loss functions and the mechanics of neural implicit scene rendering. This section also covers our evaluation criteria and the use of statistical analysis in evaluation. Following the theoretical motivation and approach, Chapter 4 presents experiments that showcase the potential for improvements in reconstruction and rendering using error-scaling. The chapter goes on to show the results of our different proposed architectures and provides a larger ablation on different datasets before exploring the multi-sensor setting. Chapter 5 presents our key takeaways from our experiments, and provides arguments for the assumptions made in our theoretical framework. Additionally, we provide some context to the challenges of learning uncertainty in an online fashion from a single sensor and present promising avenues for future work. Finally, we present our closing remarks in Chapter 6.

Chapter 2

Related Work

Visual odometry (VO) and SLAM have been an ongoing research subjects in robotics across the last four decades. The main difference between visual odometry and SLAM lies in their objective and constraints. Visual odometry is solely interested in ego-motion estimation which is equivalent to the localization in SLAM frameworks. As such, visual odometry tends to have limited memory using only information from the the past few seconds to perform motion estimation. SLAM, in contrast, aims to build a coherent map that it can localize itself within, achieving both goals simultaneously.

2.1 Visual SLAM

The first SLAM approaches formulate the problem as an Extended Kalman Filter (EKF) [38] that tracks feature points across sequences of images. Feature points detectors have evolved throughout the years with some key developments including Shi-Tomasi feature points [37], scale-invariant feature transform (SIFT) features [21], speeded up robust features (SURF) [5], and oriented FAST and rotated BRIEF (ORB) features [33, 7, 34].

To improve the efficiency of SLAM pipelines, the use of selected keyframes—that is to discard strongly overlapped intermediary frames—is a popular choice as it allows for more efficient bundle adjustment (BA) optimization. BA is a common approach for optimizing image-based reprojection error that is widely adopted across SLAM frameworks.

Another distinguishing factor in SLAM pipelines involves their optimization paradigms. SLAM methods can be divided between direct methods, which minimize photometric error of pixel intensities, or indirect methods, which calculate or detect features that are used to minimize spatial errors. Dense-SLAM methods generally rely on direct methods as they attempt to build dense maps using pixel-wise colour and depth.

For classical indirect SLAM methods, the current state-of-the-art ORB3-SLAM [8] builds off of previous ORB-SLAM [25, 26] frameworks. This method utilizes ORB features and extends the robustness with multi-map construction and re-association in the event of tracking failure.

2.2 Dense-SLAM

The foundations for dense online 3D scene reconstruction was developed by Curless and Levoy [11]. In 2012, Newcombe *et al*. build on top of this framework with KinectFusion [27], the first modern dense-SLAM pipeline using RGBD cameras. Their approach uses a globally fused volumetric model represented by a *truncated signed distance function* (TSDF). A more recent work by Cao *et al*. investigated real-time

high accuracy 3D reconstruction [9] which leverages uncertainty formulations in both spatial and depth mapping to improve global model consistency and reconstruction accuracy.

∇SLAM [18] takes classical dense-SLAM framework and re-implements many mechanisms as an end-to-end differentiable pipeline. This method takes advantage of parallel GPU processing and gradient optimization methods developed for deep learning, but does not employ learnable parameters.

2.3 Neural Implicit Representations

2.3.1 Neural Radiance Fields

In 2020, Mildenhall *et al*. presented NeRF [24], a deep approach for *novel view synthesis* using a simple multi-layered perceptron (MLP) to implicitly represent spatial volume density. A scene could then be reconstructed using volume rendering techniques [23] from arbitrary viewpoints. Since then, the simplicity of NeRF has inspired significant research activity with extensions and applications in new directions [13, 14, 15].

Oechsle *et al*. propose UNISURF [28], a 3D reconstruction framework capable of both surface and volume reconstruction by formulating implicit surface models and radiance fields in a unified way, enabling the extraction of surface and volume rendering using the same model. A key contribution involves the use of direct *occupancy* over volume density as a better constrained implicit model where the surface is represented by all 3D points lying on the occupancy level set of one half: $\mathcal{S} = \{\mathbf{x}_s | \mathrm{o}_\theta(\mathbf{x}_s) = 0.5\}$. Wang *et al*. propose NeuS [43], a surface reconstruction method defining the surface as the zero-set of an SDF: $\mathcal{S} = \{\mathbf{x}_s | \mathrm{SDF}_\theta(\mathbf{x}_s) = 0\}$. They present a novel volume rendering formulation free of bias in the first-order of approximation. Both UNISURF and NeuS enable surface reconstruction without pixel mask supervision.

In the work by Rematas *et al*., Urban Radiance Fields [31] enables 3D reconstruction and novel view synthesis by fusing RGB and lidar sweeps in urban outdoor scenes. Utilizing free space enforcement through an impulse-based surface penalty, the neural representation is encouraged to converge to more defined surface representations.

Martin-Brualla *et al*. develop NeRF-W [22], or NeRF in the Wild, an extension to NeRF that enables radiance fields on an unstructured collection of images. Key contributions include the use of a lower-dimensional latent embedding space for handling images taking under different lighting, weather, or camera conditions, and an additional uncertainty head that handles transient elements in various images.

2.3.2 Convolutional Occupancy Networks

While NeRF has proven to be extremely popular, other neural implicit representations have also been explored. Peng *et al*. propose Convolutional Occupancy Networks (CON) [29], an implicit representation that provides stronger structured reasoning and easy extension to larger scenes. CON employs a voxel feature grid that allows for interpolation of fine details. Additionally, CON represents spatial geometry via occupancy, a spatial representation that is bounded within $[0, 1]$, with values at 0.5 representing surface boundaries.

2.4 Neural Implicit SLAM

Traditional radiance fields are trained using ground truth or pre-calculated camera poses, lacking the localization capabilities required for SLAM. iNeRF [32] shows that radiance fields could be inverted to retrieve

camera poses given an image. BARF [20] is able to construct a radiance field given rough initial guesses using a hierarchical approach that procedurally added detail. Such approaches are not suited for real-time operations, but show that localization and pose correction is possible for neural implicit methods.

Inspired by NeRF, Sucar *et al*. developed iMAP [41], a SLAM framework that performs implicit mapping and positioning in real-time. iMAP showcases that an MLP can serve as the only scene representation for a SLAM system. In particular, this method is capable of efficient geometry representation with automatic detail control, as well as plausible completion of unobserved scenes due to the implicit nature of the representation.

NICE-SLAM [46] is another exploration into neural implicit SLAM, utilizing CON as opposed to radiance fields. CON has stronger structured reasoning guarantees than iMAP, preventing catastrophic forgetting through local updates to a voxel feature grid. The feature grid is also extensible for dynamic scaling to larger scenes, a process that is difficult for the fixed-size MLP representation used in NeRF-based approaches.

NeRF-SLAM [32] offers another approach for neural implicit SLAM, focusing on improving photometric accuracy and rendering. NeRF-SLAM utilizes state-of-the-art tracking from optical flow developed in DROID-SLAM [42]. Their novel contribution involves uncertainty-weighting in the training of the neural radiance field, thus improving robustness to noisy depth maps. Their mapping process simultaneously optimizes the pose and the neural parameters.

2.5 Learning-based 3D Reconstruction and SLAM

Building on top of the dense-SLAM paradigm and TSDF-Fusion originally popularized by KinectFusion [27] and other previous work [11], RoutedFusion [44] learns the signed distance update function and performs denoising and outlier correction of the input depth map. This approach is real-time capable and overcomes the challenges of hand-tuning for sensor-specific and scene-specific conditions by instead learning these characteristics a-priori.

Sandström *et al*. [35] propose SenFuNet, a deep-learning method for multi-sensor depth fusion to perform 3D reconstruction. Their pipeline is end-to-end trained in a light-weight online fashion, allowing for real-time capability. Their method locally emphasizes more accurate sensors for different scene conditions and is able to handle time asynchronous and non-rigidly mounted sensors, enabling such possibilities as multi-agent reconstruction.

The current state-of-the-art SLAM system is DROID-SLAM [42]. DROID-SLAM employs an end-to-end differentiable architecture that leverages the strengths of both traditional and deep methods. Key innovations include multi-frame handling and differentiable Dense Bundle Adjustment (DBA) that allows extensibility in the use of stereo or RGBD inputs without retraining. To achieve greater efficiency, DROID-SLAM employs custom CUDA kernels.

2.6 Uncertainty in Computer Vision

In the development of deep learning methods over the past decade, models and architectures have continued to grow in size and complexity, rendering the mechanisms for regression and classification tasks a black box. To address the explainability in deep models, producing uncertainty and quantifying confidence in predictions has become an area of active research [1].

Uncertainty can be characterized by two types of uncertainty. The first kind of uncertainty is aleatoric uncertainty, which describes the measurement uncertainty—*i.e*. uncertainty inherent to the observation due to the sensor noise. In contrast, *epistemic uncertainty* can be characterized as model uncertainty—*i.e*. the

uncertainty in the model parameters. Aleatoric uncertainty cannot be reduced with more observations as the uncertainty derives from intrinsic qualities of the input data, whereas epistemic uncertainty can be addressed by observing additional information.

Within computer vision, Kendall and Gal [19] showcase a way to model epistemic and aleatoric uncertainty in Bayesian neural networks (BNNs). In their work, they model epistemic uncertainty via Monte Carlo dropout, a variational Bayesian approximation. The aleatoric uncertainty is modeled from a second head connected to the output of the BNN and formulated implicitly via a Laplacian loss.

Bae *et al*. [2] leverage learning aleatoric uncertainty to refine surface normal predictions from RGB images. They introduce the von Mises-Fisher distribution for modeling the surface normal distribution and minimize the angular difference between the prediction and ground-truth. They use pixel-wise feature MLPs to refine the normal directions and use uncertainty-guided sampling to ensure a more balanced training dataset. This work is then applied to depth refinement in IronDepth [3], where the normal information is used to propagate depths from surrounding information based on planarity classification.

Chapter 3

Methods

3.1 NICE-SLAM Recap

NICE-SLAM, short for "Neural Implicit sCalable Encoding" SLAM, is built off of the Convolutional Occupancy Networks [29] framework for the scene representation. This approach tracks the occupancy using an encoded 3D grid of features that can be passed, after interpolation, through an MLP decoder to acquire the occupancy. Taking inspiration from NeRF [24], the approach leverages volume rendering to generate 2D depth maps, which can be directly compared with the captured depth map in the objective function. We provide a recap of key concepts in the NICE-SLAM pipeline here, an overview of which can be found in Figure 3.1.

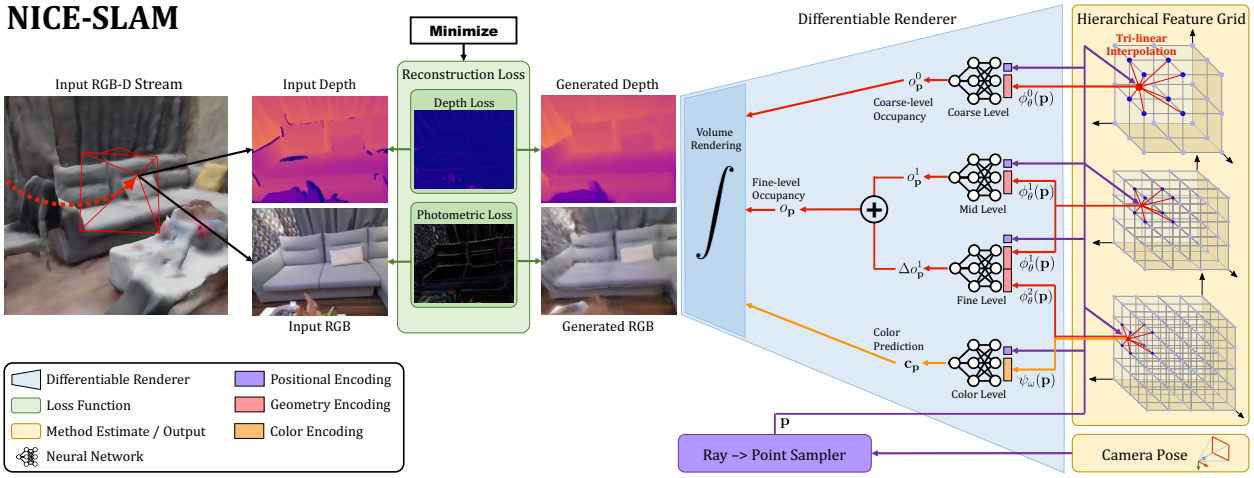


Figure 3.1: NICE-SLAM pipeline from the original paper. A stream of RGBD images (shown on the left) are passed to the pipeline, which minimizes the geometric and photometric loss with the depth and RGB images generated by NICE-SLAM via volume rendering. The scene is represented by hierarchical feature grids, where the fine-level occupancy is constructed from mid-level occupancy output and the fine-level occupancy residual. © 2022 IEEE.

3.1.1 Encoded Feature Grid

A key differentiator between iMAP and NICE-SLAM is their choice of scene encoding. iMAP leverages a NeRF-like representation where the scene is encoded in the weights of an MLP. In contrast, NICE-SLAM encodes the scene in hierarchical voxel grids of features. For any sampled point, a feature from these voxel grids can be tri-linearly interpolated from its neighbours. These features can then be fed, in a coarse-to-fine manner, through their respective decoder to extract the occupancy probability of the given point.

The geometry is encoded in three feature grids—coarse, middle, and fine. Each feature grid ϕ_θ^l has an associated decoder f^l, where $l \in \{0, 1, 2\}$. Additionally, the colour is encoded in a fourth feature grid ψ_ω and decoder g_w, used for further scene refinement after initial stages of geometric optimization. The observed scene geometry is reconstructed from the middle and fine resolution feature grids, with the fine feature grid output being added to the middle grid occupancy in a residual manner. The standard geometric decoder MLP architecture employed in NICE-SLAM is detailed in Figure 3.2.

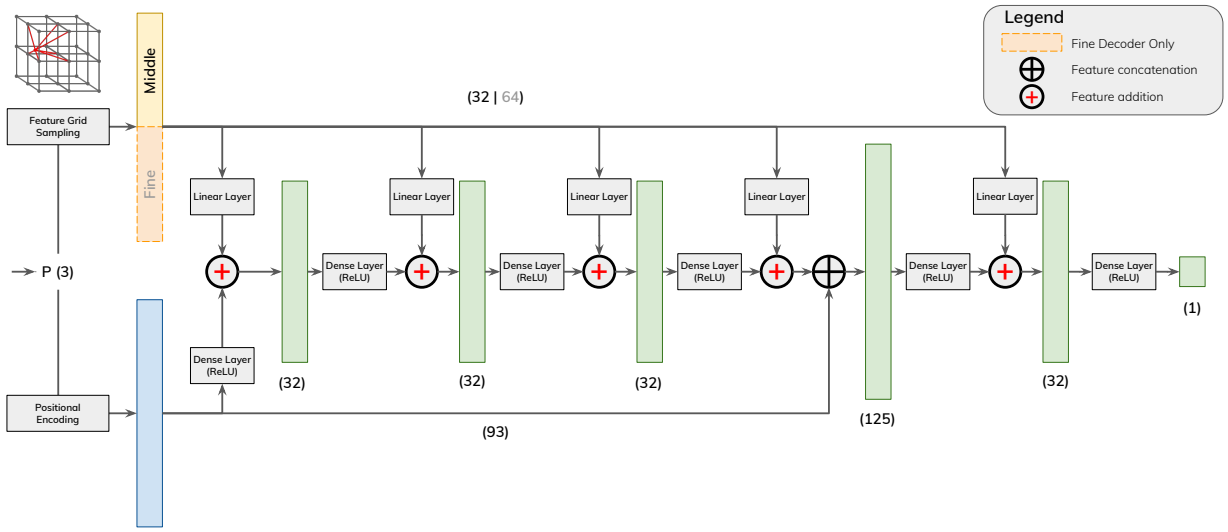


Figure 3.2: The standard geometric MLP architecture used in NICE-SLAM.

The colour decoder is similar in construction, but outputs three values for RGB information and does not utilize fine features.

3.1.2 Volume Rendering

NeRF and NICE-SLAM's approach to scene rendering leverages volume rendering for occlusion-aware generation of depth maps and RGB images. This process involves sampled points $\mathbf{p}_i \in \mathbb{R}^3$ from depth $d_i \in \mathbb{R}^1$ along a ray $\mathbf{r} \in \mathbb{R}^3$ cast from origin $\mathbf{O} \in \mathbb{R}^3$, as detailed in Eq. (3.1).

$$\mathbf{p}_i = \mathbf{O} + d_i \mathbf{r}, \quad i \in \{1, ..., N\} \tag{3.1}$$

The coarse-level occupancy $o^c_{\mathbf{p}_i}$ and fine-level occupancy $o^f_{\mathbf{p}_i}$ are the outputs of our decoder networks f^l. The colour output is produced by the decoder g_w. We show how the different outputs of the network are used to generate occupancies and colours in Eq. (3.2).

$$o^c_{\mathbf{p}_i} = f^0(\mathbf{p}_i) \quad o^f_{\mathbf{p}_i} = f^1(\mathbf{p}_i) + f^2(\mathbf{p}_i) \quad \mathbf{c}_i = g_w(\mathbf{p}_i) \tag{3.2}$$

From these occupancies along the ray, volume rendering constructs a weighting function, w_i, based on the termination criteria in alpha-compositing occupancies $o_{\mathbf{p}}$ as described in Eq. (3.3). This is done for both the coarse-level and fine-level occupancies. This weight represents the discretized probability that the ray would terminate at that particular point.

$$w^c_i = o^c_{\mathbf{p}_i} \prod_{j=1}^{i-1} (1 - o^c_{\mathbf{p}_j}), \quad w^f_i = o^f_{\mathbf{p}_i} \prod_{j=1}^{i-1} (1 - o^f_{\mathbf{p}_j}) \tag{3.3}$$

To supervise NICE-SLAM, we require a depth map and RGB image rendering to compare against the captured sensor data. To get the associated depth from the weighting function described previously, we take the weighted average of the depth values along each ray. A similar approach extracts the colour from the feature grid through the decoder g_w. These approaches are described in Eq. (3.4).

$$\hat{D}^c = \sum_{i=1}^{N} w^c_i d_i, \quad \hat{D}^f = \sum_{i=1}^{N} w^f_i d_i, \quad \hat{I} = \sum_{i=1}^{N} w^f_i \mathbf{c}_i \tag{3.4}$$

This volume rendering method also provides us a variance from the discretized selection of points used in the process. By taking the depth differences multiplied by the weighting function, we can extract a variance that is a composite of the model uncertainty and sampling uncertainty, as shown in Eq. (3.5).

$$\hat{S}^c_D = \sum_{i=1}^{N} w^c_i \left(\hat{D}^c - d_i\right)^2, \quad \hat{S}^f_D = \sum_{i=1}^{N} w^f_i \left(\hat{D}^f - d_i\right)^2 \tag{3.5}$$

3.1.3 Mapping

The mapping process in NICE-SLAM utilizes an L1 loss between the rendered ($\hat{D}_m$) and captured depth maps (D_m), as well as the rendered ($\hat{I}_m$) and captured RGB images (I_m). These parameters are combined through a weighting variable, λ_{pm}. In implementation, M pixels, or rays, are sampled from the RGBD images to reduce computation and mimic the approach used in stochastic gradient descent (SGD). The loss functions are detailed in Eqs. (3.6) to (3.8).

$$\mathcal{L}_{\text{map}} = \mathcal{L}^c_g + \mathcal{L}^f_g + \lambda_{pm}\mathcal{L}_p \tag{3.6}$$

$$\mathcal{L}^l_g = \frac{1}{\text{M}} \sum_{m=1}^{M} |D_m - \hat{D}^l_m|, \quad l \in \{c, f\} \tag{3.7}$$

$$\mathcal{L}_p = \frac{1}{\text{M}} \sum_{m=1}^{M} |I_m - \hat{I}_m| \tag{3.8}$$

We note that the equations have been fully reproduced from the original paper. In the official code release, the coarse mapper acts as a separate process and is refined separately from the middle-fine mapper.

3.1.4 Tracking

The tracking loss in NICE-SLAM scales the L1 difference by the depth variance calculated from the variance in the weighted occupancy. The loss is also formulated through a combination of geometric and photometric losses controlled by a weight parameter λ_{pt}, and described in detail in Eqs. (3.9) and (3.10). The original NICE-SLAM paper describes using both the coarse and fine depth renderings in its loss function to perform short-range predictions on scene geometry.

$$\mathcal{L}_{\text{track}} = \mathcal{L}_{g,var} + \lambda_{pt}\mathcal{L}_p \tag{3.9}$$

$$\mathcal{L}_{g,var} = \frac{1}{M_t}\sum_{m=1}^{M_t} \frac{|D_m - \hat{D}_m^c|}{\sqrt{\hat{S}_D^c}} + \frac{|D_m - \hat{D}_m^f|}{\sqrt{\hat{S}_D^f}} \tag{3.10}$$

Despite the claims in the original paper, the officially released code only uses the fine resolution rendering for its tracking loss function.

3.2 Uncertainty-aware Loss

The previous section details how the original NICE-SLAM constructs its loss functions and the general concepts employed throughout its SLAM pipeline. We aim to introduce uncertainty-awareness to the NICE-SLAM framework to improve robustness against sensor or aleatoric noise, which can be a product of the sensor itself, or the processing used to generate the depth map.

We motivate our formulation of sensor noise under the assumption of a Gaussian noise distribution on a per-ray basis. That is, each pixel m in the captured sensor data is treated independently. Consequently, the measured depth is sampled from a probability distribution described in Eq. (3.11).

$$P(D_m) = \frac{1}{\sqrt{2\pi\beta_m^2}} e^{-\frac{(D_m-\mu_m)^2}{2\beta_m^2}} \tag{3.11}$$

We take μ_m to be the true depth and β_m to be the standard deviation of the depth reading of a specific pixel. When we aggregate all depth sensor information, we get the joint density of the per-ray depth observations. As previously mentioned, we assume the error in each per-pixel depth sensor to be independent and identically distributed (I.I.D.). The joint distribution is described in Eq. (3.12).

$$\begin{aligned} P(D_1, ..., D_M) &= P(D_1)...P(D_M) \\ &= \prod_{m=1}^{M} P(D_m) \\ &= \prod_{m=1}^{M} \frac{1}{\sqrt{2\pi\beta_m^2}} e^{-\frac{(D_m-\mu_m)^2}{2\beta_m^2}} \end{aligned} \tag{3.12}$$

The best estimate of the depth can thus be determined via maximum likelihood estimation (MLE). The MLE is described in Eq. (3.13).

$$\begin{aligned} \arg\max_{\theta_m} P(D_1, ..., D_M|\theta_m) &= \arg\min_{\theta_m} -\log(P(D_1, ..., D_M|\theta_m)) \\ &= \arg\min_{\theta_m} \sum_{m=1}^{M} \frac{(D_m-\mu_m)^2}{2\beta_m^2} + \frac{1}{2}\log(\beta_m^2) \end{aligned} \tag{3.13}$$

where θ_m represents the parameters that define the mean and standard deviation of the distribution.

As previously discussed, the original implementation renders the per-pixel depth and its volume-rendered variance. This could naively be applied to the previous equations with the rendered depth $\hat{D}_m$ representing μ and the variance $\hat{S}_D$ representing β^2. Unfortunately, such an approach is poorly motivated as this calculated variance is related to the model confidence, as opposed to the sensor-specific noise. In practice, the uncertainty we strive to model is aleatoric uncertainty and should be distinct from the model confidence.

One interpretation of this variance calculated from the volume rendering is as the epistemic uncertainty. With an increasing number of observations, the epistemic uncertainty should shrink, driving the model towards sharp bounds, an attribute that is less affected by the sampling and rendering biases described further in Appendix B.

We instead want to have a separate process to extract aleatoric uncertainty. We take the concept of implicitly learned aleatoric uncertainty from the work of Kendall and Gal [19]. We investigate multiple

methods to generate uncertainty, including the use of an analogous volume-rendered approach, a simple per-pixel MLP, and a patch-based MLP. These approaches take in spatial information from the specific image frame, and potentially from the scene encoding, to generate uncertainty $\hat{\beta}$, distinct from the rendered variance $\hat{S}_D$. The probability formulation and objective function thus substitute μ and β with $D_m(\phi_m)$ and $\hat{\beta}(\xi_m)$, as shown in Eq. (3.14). The depth model is parameterized by ϕ and the uncertainty model is parameterized by ξ. We thus perform MLE by minimizing the arguments $\theta = \{\phi, \xi\}$.

$$\arg\max_{\phi_m,\xi_m} P(D_1, ..., D_M|\phi_m, \xi_m) = \arg\min_{\phi_m,\xi_m} \sum_{m=1}^{M} \frac{(D_m - \hat{D}_m)^2}{2\hat{\beta}_m^2} + \frac{1}{2}\log(\hat{\beta}_m^2) \tag{3.14}$$

We ensure that our occupancy network is decoupled from the aleatoric uncertainty model to prevent adverse effects on the model confidence. This formulation permits the model uncertainty to be driven down to reduce the effects of weighting function bias described in Appendix B, while enabling the implicit learning of the variance parameters. In practice, we assume a Laplacian distribution, corresponding to an L1 loss, which was found by Kendall and Gal [19] to perform better on vision tasks. The probability formulation is described in Eq. (3.15).

$$\begin{aligned} P(D_1, ..., D_M) &= P(D_1)...P(D_M) \\ &= \prod_{m=1}^{M} P(D_m) \\ &= \prod_{m=1}^{M} \frac{1}{2\beta_m} e^{-\frac{|D_m - \mu_m|}{\beta_m}} \end{aligned} \tag{3.15}$$

The corresponding best estimate via maximum likelihood estimation (MLE), given a parameterized model of depth and uncertainty, can be found in Eq. (3.16) The formulated loss objective is shown in Eq. (3.17).

$$\begin{aligned} \arg\max_{\phi_m,\xi_m} P(D_1, ..., D_M|\phi_m, \xi_m) &= \arg\min_{\phi_m,\xi_m} -\log(P(D_1, ..., D_M|\phi_m, \xi_m)) \\ &= \arg\min_{\phi_m,\xi_m} \sum_{m=1}^{M} \frac{|D_m - \hat{D}_m(\phi_m)|}{\hat{\beta}_m(\xi_m)} + \log(\hat{\beta}_m(\xi_m)) \end{aligned} \tag{3.16}$$

$$\mathcal{L} = \sum_{m=1}^{M} \frac{|D_m - \hat{D}_m(\phi_m)|}{\hat{\beta}_m(\xi_m)} + \log(\hat{\beta}_m(\xi_m)) \tag{3.17}$$

3.3 Modifications to Uncertainty-aware Loss

In Section 3.2, we provided an argument for constructing a loss function that implicitly extracts uncertainty under the supervision of noisy depth maps. Taking inspiration from other works that utilize uncertainty for rendering 3D scenes, we propose two potential additions.

The first modification was introduced in NeRF-W [22], where they enforce a minimum uncertainty value $\beta_{\min}$. As done in their work, we apply softplus activation to the output $\tilde{y}_m$ of our network, thus bounding $\tilde{y}_m$ within $(0, \infty)$. The addition of a minimum uncertainty —or a "minimum importance" factor—consequently changes the bound of the uncertainty to $(\beta_{\min}, \infty)$. Zero bounds can generate unstable results as some losses are heavily blown up do to division by zero, and adding this factor improves training stability. The softmax and minimum importance factor can be seen in Eq. (3.18).

$$\hat{\beta}_m = \beta_{\min} + \log\left(1 + \exp\left(\tilde{y}_m\right)\right) \tag{3.18}$$

The second alternative modification is to transform the loss function from uncertainty to *confidence*. That is, we have some confidence $\hat{C}$ in the depth reading, bounded within $(0, 1)$, achievable using a logistic activation function. Practically, this enforces an uncertainty with the bounds of $(1, \infty)$. This results in a loss function shown in Eq. (3.19).

$$\mathcal{L}_g = \sum_{m=1}^{M} \hat{C}|D_m - \hat{D}_m| - \log(\hat{C}) \tag{3.19}$$

This loss function is similar to the one used in RoutedFusion [44]. One addition Weder *et al*. contribute to this formulation is a scaling factor λ_c on the log term, which controls the expressivity or range of the confidence values, modifying the loss function as shown in Eq. (3.20).

$$\mathcal{L}_g = \sum_{m=1}^{M} \hat{C}|D_m - \hat{D}_m| - \lambda_c \log(\hat{C}) \tag{3.20}$$

3.4 Uncertainty Network Architecture

NICE-SLAM employs a hierarchical training scheme in mapping, refining the scene geometry in a coarse-to-fine manner. More specifically, the mapping process runs through three distinct stages. The first stage is the middle-grid refinement where rough geometry can be learned. The second stage of refinement begins to jointly optimize the fine-grid and middle-grid. The output of the fine-grid decoder is added to the output of the middle-grid decoder in a residual fashion. The final stage incorporates the colour-grid and decoder refinement, providing photometric consistency to the depth refinement process.

In this work, we first evaluate results using only depth maps in a single-sensor. As such, the second and final stage are identical in our training pipeline as we do not perform colour refinement. We additionally retain the first stage loss (middle-grid only) using the simple depth difference, allowing for easier learning of coarse geometry. We provide details on extending this method to a two-sensor environment in Section 3.5.

3.4.1 Uncertainty via Grid of Features

Capturing uncertainty in an analogous fashion to the depth and colour is a natural extension under the NICE-SLAM framework. We present the architectural changes necessary for this approach in Figure 3.3.

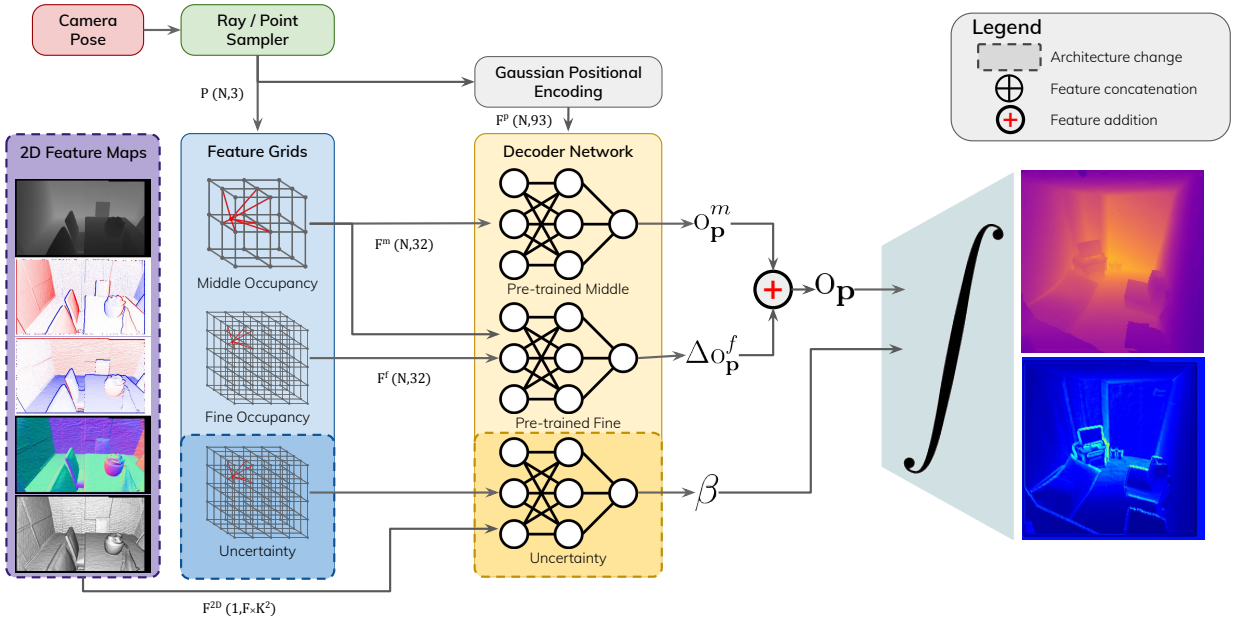


Figure 3.3: Volume-rendered uncertainty NICE-SLAM pipeline. The additions to the original pipeline are bordered by dashed lines. We have an additional grid of features for uncertainty and an additional decoder. The uncertainty network takes in a subset of 2D feature map information on a per-ray basis to provide additional useful information for calculating uncertainty.

This volume-rendered approach is similar to the approach taken in NeRF-W [22], where they get an uncertainty value at each sampled point along the ray and render the uncertainty using the weighting function, similar to Eq. (3.4). In this approach, we append an additional feature grid for uncertainty and initialize an associated decoder h_w. Equation (3.22) show the uncertainty rendering equations.

$$b_i = h_w(\mathbf{p}_i) \tag{3.21}$$

$$\hat{\beta} = \sum_{i=1}^{N} w_i^f b_i \tag{3.22}$$

The feature grid and its associated decoder are implicitly trained throughout the fine and colour stages of the optimization process using the modified loss function as described in Eq. (3.17). This approach allows for aggregation of uncertainty information across frames. Knowledge of uncertain regions, such as edges, can be propagated from one frame to the next.

However, estimating the aleatoric or sensor uncertainty solely from points sampled in the 3D scene fails to capture the per-pixel information of the image representation. We can instead use both the observed 3D scene and pixel-wise information extracted from 2D feature maps. We explain these 2D feature maps in more detail in Section 3.4.2. We append the informative features from the 2D feature maps, alongside per-point depth in the ray sampling, to the positionally encoded points. These features are then fed as input into the decoder network to output the per-pixel uncertainty for each sampled ray using the volume rendering approach from the point occupancy weighting.

3.4.2 Uncertainty via Ray-based MLP

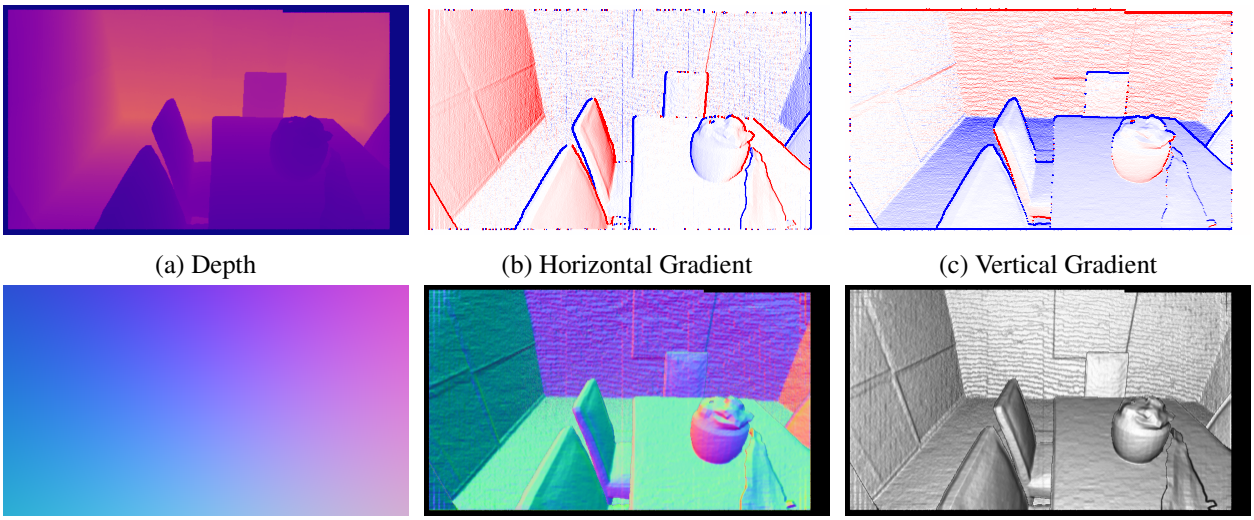
In the previous section, we presented a natural extension to NICE-SLAM using a 3D grid of features. Such an approach aggregates information over frames. However, this approach is dependent on the camera pose for the scene rendering and reliant on the model occupancy for volume rendering. These aspects naturally couple the aleatoric and epistemic uncertainty of the SLAM pipeline, which may limit the utility of the output uncertainty.

A specific long-term goal of this project is to extend the results to the multi-sensor configuration. Within that environment, we wish for the uncertainty to be decoupled from the epistemic uncertainty of the 3D scene, allowing for us to balance the influence of both sensors. An additional concern within the NICE-SLAM framework is the computational overhead. The original implementation only uses a sparse sub-selection of pixels, or rays, to allow for real-time capable operation. Volume rendering is one of the more expensive operations within NICE-SLAM and an additional rendering for each sensor may be prohibitively expensive.

Consequently, we propose a simpler approach to derive a ray-specific uncertainty through the use of 2D feature maps that contain potentially important features. We can leverage cheaply available metadata, as was done in SimpleRecon [36], to capture sensor noise from the 2D feature maps generated from the depth readings. We investigate plausible per-pixel (per-ray) features which include:

1. $D_m \in \mathbb{R}^1$, the measured depth value
2. $\mathbf{N} \in \mathbb{R}^3$, the surface normal direction in camera frame
3. $\mathrm{dx} \in \mathbb{R}^1$, the horizontal gradients of D_m
4. $\mathrm{dy} \in \mathbb{R}^1$, the vertical gradients of D_m
5. $\theta \in \mathbb{R}^1$, the incident angle between local ray and surface normal

The above features are fed through a simple MLP network h_w^2, similar in construction to the MLPs used for the occupancy decoder in vanilla NICE-SLAM. We use a network with 5 intermediate layers with 32 nodes each, activated via ReLU. Figure 3.4 shows the different features used as inputs to the MLP to leverage the 2D information available in the image.



(a) Depth (b) Horizontal Gradient (c) Vertical Gradient

(d) Local Ray Direction (e) Surface Normal (f) Incident Angle

Figure 3.4: 2D feature maps used for pixel-wise ray features. (Top) D_m, dx, dy (Bottom) local, $\mathbf{N}$, θ

3.4.3 Uncertainty via Patch-based MLP

In Section 3.4.2, we proposed a simple MLP network trained to derive uncertainty from a set of cheaply available metadata. Such an approach is notably low-dimensional, which may lead to poor results without additional local context.

We propose a natural extension to the previous method by using a region-based expansion of the receptive field of the ray. That is, we expand the ray kernel to a 5×5 patch, increasing the number of input features by a factor of 25. This patch of pixels gives local context and local correlation of uncertainty for areas near edges or with high frequency features. To illustrate the difference in coverage, we show the 1×1 ray selection vs. the 5×5 patch selection in Figure 3.5.

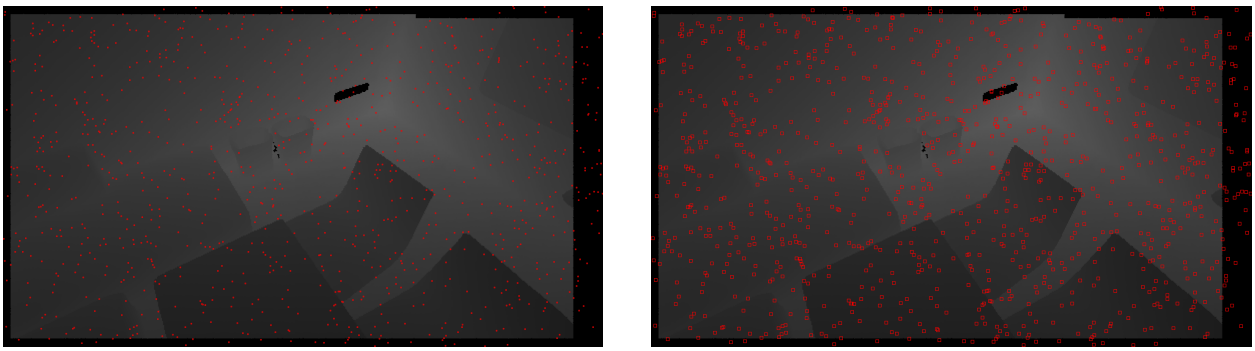


Figure 3.5: Visual comparison between the coverage using a single pixel compared to a patch of pixels.

In implementation, the ray and patch-based approaches are closely coupled and we will refer to these as the 2D MLP approach. We showcase an overview of the the data flow in Figure 3.6. This is distinguished from the volume-rendered approach which captures accumulated understanding of uncertainty via a volumetric grid of features.

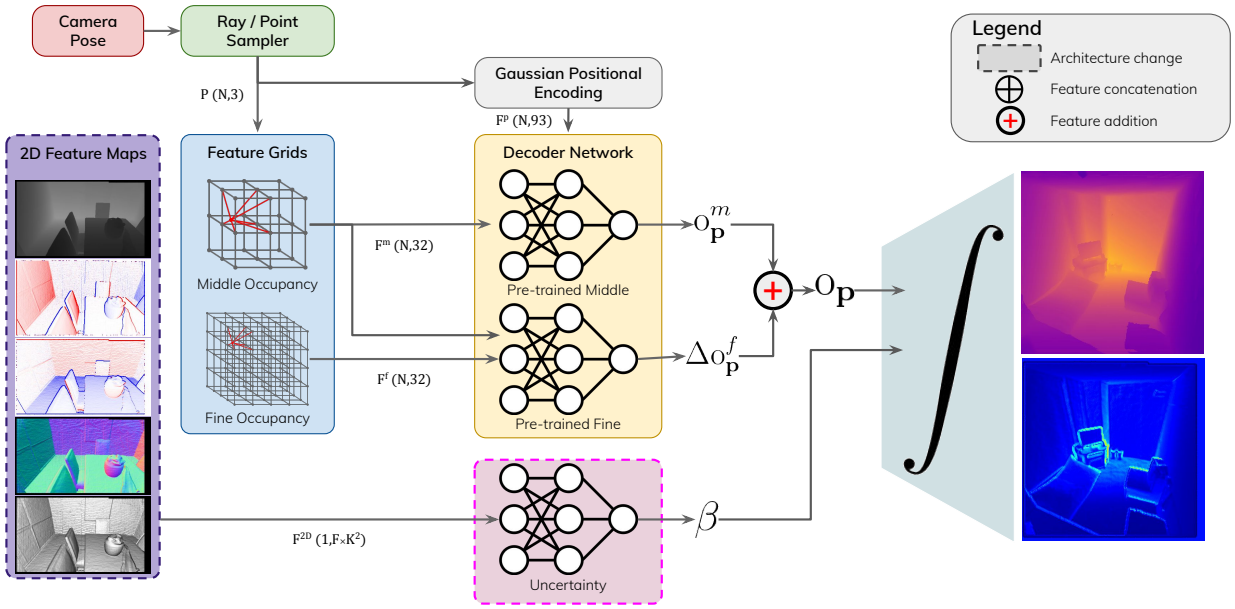


Figure 3.6: 2D MLP uncertainty NICE-SLAM pipeline. The additions to the original pipeline are bordered by dashed lines. We have an additional MLP for the uncertainty estimation in pink. The uncertainty network takes in a subset of 2D feature map information on a per-ray basis to provide additional useful information for calculating uncertainty.

In contrast to the 3D approach, or volume-rendered approach, the 2D MLP approach does not aggregate information across frames, relying only on the current frame for information to extract uncertainty. The MLP h_w^2 is not volume rendered across the scene, allowing for better efficiency and access to the estimated uncertainty directly from the set of features ζ, as shown in Eq. (3.24).

$$\hat{\beta} = h_w^2(\zeta) \tag{3.23}$$

(3.24)

The extracted features $zeta$ are the input features listed above for each pixel in the 1×1 or 5×5 patch.

3.5 Aligned Two-sensor Extension

The methods described so far have encompassed implicitly learning an uncertainty given a single sensor. For any given pixel, this approach only has a single observation. We extend this single-sensor approach to incorporate a second sensor, which we assume to be aligned—*i.e.* in the same frame of reference—and synchronized. If we again assume that each depth observation is I.I.D., the joint likelihood we wish to maximize is the product of the probability distributions for each pixel in each sensor.

Given two synchronized and aligned sensors, we can sample a set of pixels $m \in \{1, ..., M\}$ from depth sensor A and depth sensor B. The corresponding generalized loss function is shown in Eq. (3.25).

$$\mathcal{L} = \sum_{m=M}^{M} \left(\frac{|D_{m,A} - \hat{D}_{m,A}|}{\hat{\beta}_{D,m,A}} + \log(\hat{\beta}_{D,m,A}) + \frac{|D_{m,B} - \hat{D}_{m,B}|}{\hat{\beta}_{D,m,B}} + \log(\hat{\beta}_{D,m,B}) \right) \tag{3.25}$$

One interpretation of this objective function is that the pipeline implicitly learns the weighting between these two sensor observations. The loss function penalizes large uncertainties via the log terms, and implicitly learns the uncertainty for both sets of observations as the model depth is optimized.

To accommodate the two-sensor extension, we make a few modifications to the original code to aggregate and combine depth information. These changes are required to ensure that each sensor's information is equally utilized throughout the NICE-SLAM pipeline. We detail these changes in Appendix D.

Chapter 4

Experiments and Results

4.1 Evaluation Criteria

We use the same metrics as presented by iMAP [41] and NICE-SLAM [46]. These errors can be divided between:

1. **Tracking Error.** We use the absolute trajectory error (ATE) RMSE [40] to compare tracking error across methods. This error normally computes the translational difference of the track after least-squares alignment. We have modified this error to be computed without least-squares alignment to better analyze drift, as the initial pose is fixed at the ground-truth pose.

2. **3D Metrics (Reconstruction).** The 3D metrics are evaluated using: *Accuracy (*cm*)*—the mean distance from a point in the generated mesh to the ground-truth mesh; *Completion* (cm)—the mean distance of the ground-truth mesh to the generated mesh; and *Completion Ratio* (< 5 cm %)—the percentage of points in the ground-truth mesh that have a point within 5 cm of the generated mesh.

3. **2D Metric (Rendering).** The average L1 depth loss between 1000 randomly rendered depth maps from the reconstructed and ground-truth meshes.

As noted in Appendix A.3, NICE-SLAM produces stochastic results for each run, even when random number generator seeds are provided. As such, single run results cannot be directly compared as the difference in performance may be simply attributed to the stochasticity of the program. To address this issue, we perform repeated tests and report mean results for greater robustness to the stochasticity inherent to NICE-SLAM.

We also employ unpaired—also known as independent sample or Welch's—t-tests [45] to evaluate the significance of the difference in results. As each of our runs is independent of other runs, we compare using an unpaired assumption. Additional assumptions for an unpaired t-test include:

- The NICE-SLAM outputs are randomly sampled and reflective of the true distribution.
- The true performance is approximately normal in its distribution.
- There should be no extreme outliers in either set of samples.

The unpaired t-test is a two sample location test that compares if two sample populations have the same mean. This analysis is completed by determining the statistic t and the degrees-of-freedom ν. Given the sample means $\overline{X}_{\{1,2\}}$ and the standard errors $s_{\bar{X}_{\{1,2\}}}$, t and ν can be calculated using Eqs. (4.1) and (4.2).

$$t = \frac{\Delta\overline{X}}{s_{\Delta\bar{X}}} = \frac{\overline{X}_1 - \overline{X}_2}{\sqrt{s^2_{\bar{X}_1} + s^2_{\bar{X}_2}}} \tag{4.1}$$

$$\nu \approx \frac{\left(\frac{s_1^2}{N_1} + \frac{s_2^2}{N_2}\right)^2}{\frac{s_1^4}{N_1^2\nu_1} + \frac{s_2^4}{N_2^2\nu_2}} \tag{4.2}$$

These values can then be used to identify the probability given by Student's t-distribution that the two sample means are equal. If we assume significance at $P < 0.05$, we can determine if an improvement—i.e. an increase in the mean performance—should be considered significant.

4.2 Datasets

In evaluating the performance of our SLAM pipeline, we explore both synthetic and real datasets. We compile a summary of datasets we consider in Table 4.1.

Table 4.1: Summary of different datasets used to evaluate our method.

Source	Dataset	Type	Sensor Technology	Release
TUM	TUM-RGBD [40]	Real	Structured Light	2012
ETH-CVL	SFN-Replica [35]	Replica [39]	Structured Light (Synthetic)	2022
ETH-CVL	SFN-Replica [35]	Replica [39]	SGM Stereo	2022
ETH-CVL	SFN-Replica [35]	Replica [39]	PSMNet	2022
ETH-CVL	NS-Replica v2	Replica [39]	Structured Light (Synthetic)	2022

The Replica Dataset [39], developed by Meta Reality Labs (formerly Facebook Reality Labs) in 2019, offers a highly photo-realistic synthetic environment for 3D reconstruction and SLAM tasks. This environment includes single rooms, office spaces, and larger multi-room apartment spaces. As this environment can be synthetically generated, there are no errors in the the ground truth meshes and annotations. We note there is negligible quantization error based on the depth scale of the depth maps.

Within the synthetic Replica environment, custom trajectories, custom camera models, and custom sensor configurations can be used to generate custom datasets. iMAP [41] and NICE-SLAM [46] provide a selection of custom datasets that include office scenes 0-4 and room scenes 0-2. These datasets consist of 2000 frames employing a resolution of 1200×680, a horizontal FoV of $90°$, and a single RGB-D sensor. SenFuNet [35] provides a dataset containing scenes from offices $\{0,1,3,4\}$, rooms $\{0,2\}$, hotels $\{0\}$, apartments $\{1\}$, and frl_apartments 0-1. Each scene has several trajectories ranging from $\sim$300 to $\sim$3000 frames employing a resolution of 512×512, a horizontal FoV of $90°$, and a stereo pair of RGB-D sensors. The SenFuNet dataset further processes the depth maps using a structured light sensor noise model [4, 16], and provides dense depth maps using stereo-based semi-global matching (SGM) [17] and PSMNet [10]. These depth representations reflect highly common sensors used for depth ranging. Example depth maps from SFN-Replica are shown in Figure 4.1.

The SL noise model involves random offsets to shift pixel locations in the image and bilinearly interpolates the ground truth depth values to simulate SL sensor noise. The shifts induce noise into the depth maps while the bilinear interpolation ensures local correlation for the depth map. Subsequently, the depth values are converted to disparity where I.I.D. Gaussian noise is stacked on top of the depth values. Quantization rounding is then applied prior to converting disparity back into depth values. A more detailed explanation can be found in [4, 16].

NS-Replica is a dataset generated with only a single RGBD sensor and without any noise. We induce noise into this dataset using the same noise model as in SFN-Replica, but scaled to the dimensions of the NS-Replica images. This provides us with another synthetic dataset, but with a much smoother trajectory compared to SFN-Replica. We also note that this dataset has much finer depth resolution, with every meter represented by an intensity difference of 6533.5 compared to 1000 in SFN-Replica.

Nevertheless, the sim2real gap is a known phenomenon that impedes models trained on synthetic data from operating in the real world, even with synthesized noise models. Consequently, testing and validating on real-world datasets is required to show that performance can be transferred into practical operating settings. An older dataset is the TUM-RGBD Dataset [40]. This dataset captured data using a Microsoft Kinect, which leverages SL methods for depth extraction. Some additional datasets that are not explored in

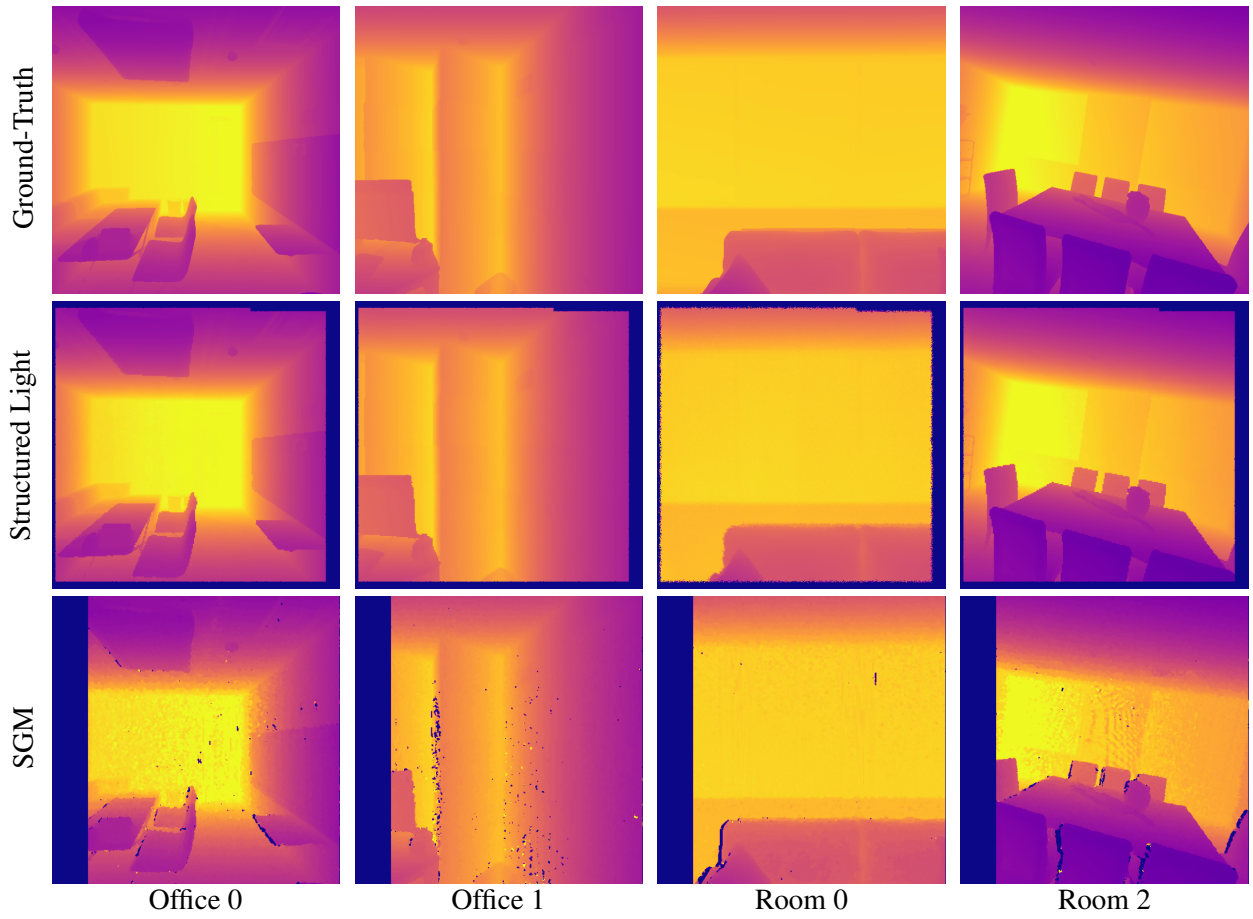


Figure 4.1: (Top) GT depth maps. (Middle) SL depth maps. (Bottom) SGM Stereo depth map. Replica scenes rendered from the SFN-Replica dataset.

this work include ScanNet [12], which also uses an SL-based sensor, and the apartment scene captures with an Azure Kinect v2 RGBD camera, which uses a ToF sensor, recorded for NICE-SLAM [46].

4.3 Implementation Details

We leave many of the hyperparameters from vanilla NICE-SLAM as is. We retain the standard grid size of $0.32\ \mathrm{m}$ for the middle feature grid and $0.16\ \mathrm{m}$ for the fine feature grid on the Replica datasets. We retain the standard grid size of $0.16\ \mathrm{m}$ for the middle feature grid and $0.08\ \mathrm{m}$ for the fine feature grid on the TUM-RGBD dataset. The ray sampling strategy remains the same, with 32 points uniformly sampled along the ray and 16 points sampled uniformly near the depth reading. The feature grids store 32-dimensional features that are interpolated for the point sampling and passed into the standard NICE-SLAM architecture.

We leave the learning rates for feature grid optimization under the same schedule—*i.e.* 0.1 for the middle stage and 0.005 for the fine stage. The "bundle adjustment" camera pose learning rate remains the same (0.001) when tracking is enabled, but is set to 0 when we run under mapping-only conditions. The decoder learning rate is turned on and set to 0.005 during the fine stage. The tracking learning rate for the camera pose is set to 0.001. We set the colour weighting parameters, λ_{pm} and λ_{pt}, to 0, or equivalently the terms are removed from the loss function.

The dataset specific parameters are specified in Table 4.2. These parameters were not tuned and may be optimized to further improve performance. Specifically, the learning rates may be adjusted under the new loss formulation to improve stability.

Table 4.2: Parameter configurations for each dataset, including the interval between mapping steps, the # of sampled pixels, # of iteration steps, and the refinement stage transition point. **Tr.**: Tracking **Its**: Iterations (optimization steps) **Trans.**: Transition (from middle-only to middle + fine stage)

Dataset	Map Interval	# Tr. Px	# Map Px	# Tr. Its	# Map Its	# 1st Map Its	Fine Trans.
TUM-RGBD	1	5000	5000	200	60	1500	40%
SFN-Replica	5	5000	5000	10	60	1500	40%
NS-Replica v2	5	5000	5000	10	60	1500	40%

We also note that the official code release does not use the coarse scene geometry in its tracking or mapping loss. As such, we run NICE-SLAM without the coarse mapper for our experiments.

We also introduce various new parameters using the uncertainty-aware loss function. These parameters include which architecture to use, the choice of minimum importance factor $\beta_{\min}$, the patch size, and the selection of input features. The different parameter choices are discussed in Section 4.5.

4.4 Evaluating with Ground-truth Data

4.4.1 Ground-truth Depth

We compare the results of 3D reconstruction performance without tracking using the different noise models with the SFN-Replica dataset. We find that the SL model has the best performance against using the ground truth depth maps, followed by PSMNet stereo rendering and SGM stereo rendering respectively. The mean metrics can be found in Table 4.3, with the standard deviation in parentheses. We refer to the original implementation of NICE-SLAM as "OG" in the following tables in this section. We also track the number of trials performed in each table under the column "N".

Table 4.3: 3D Reconstruction evaluation metrics comparing the original loss acting on depth maps with different noise models in SFN-Replica.

Scene	Sensor	Tr. Loss	Map Loss	N	ATE [m]	Acc. [cm]	Comp. [cm]	C. Ratio [%]	2D L1 [cm]
Office 0	GT	-	OG	10	-	2.38 (0.08)	1.73 (0.02)	96.4 (0.2)	1.35 (0.03)
Office 1	GT	-	OG	10	-	2.57 (0.27)	1.53 (0.04)	96.8 (0.3)	1.21 (0.04)
Room 2	GT	-	OG	10	-	1.81 (0.03)	1.73 (0.01)	97.1 (0.2)	1.25 (0.02)
Office 0	SL	-	OG	15	-	3.03 (0.17)	2.08 (0.02)	94.5 (0.2)	1.80 (0.03)
Office 1	SL	-	OG	15	-	2.71 (0.23)	1.66 (0.06)	96.1 (0.4)	1.37 (0.03)
Room 2	SL	-	OG	15	-	2.80 (0.09)	2.39 (0.02)	92.6 (0.3)	2.11 (0.02)
Office 0	SGM	-	OG	15	-	13.19 (0.83)	2.56 (0.03)	91.1 (0.3)	2.39 (0.04)
Office 1	SGM	-	OG	15	-	44.19 (3.09)	3.00 (0.04)	88.0 (0.3)	3.93 (0.11)
Room 2	SGM	-	OG	15	-	14.03 (1.72)	2.80 (0.05)	89.6 (0.5)	3.03 (0.05)
Office 0	PSM	-	OG	10	-	3.27 (0.14)	2.57 (0.03)	89.9 (0.2)	2.07 (0.03)
Office 1	PSM	-	OG	10	-	4.14 (0.32)	2.58 (0.02)	87.9 (0.3)	3.29 (0.39)
Room 2	PSM	-	OG	10	-	3.95 (0.14)	2.66 (0.03)	89.3 (0.2)	2.38 (0.03)

As one might expect, we see significant degradation of the reconstruction performance and rendering results when we introduce noise into the original NICE-SLAM pipeline. The performance of NICE-SLAM using the noiseless input depths represents an upper bound on the performance we may expect with our uncertainty-aware modifications. We hope that by learning and scaling by uncertainty, we can approach the results of using the ground-truth depth maps.

4.4.2 Proxy Uncertainty

Using an error map (absolute difference between ground-truth depth and noisy depth) as a proxy uncertainty, we get a representative aleatoric uncertainty. This approach offers a more conservative upper bound on the possible performance improvement with our learned uncertainty. In Figure 4.1, we can see both the original depth map and a noise-injected depth map from the SL noise model for the SFN-Replica dataset. Figure 4.2 showcases the absolute difference between two such depth maps and the signed difference between the two methods as rendered in the NS-Replica dataset. We note that depth differences are most noticeable at discontinuities, at further distances, and with relation to radial lens distortion effects.

We run multiple baseline tests using the proxy uncertainty as a scaling factor in comparison to the original loss function. The original loss functions are detailed in Section 3.1.3 for mapping and Section 3.1.4 for tracking. We employ modified loss functions, with the proxy "ground-truth" error acting as β_m. The modified loss functions are akin to those found in Section 4.4.2, using the L1 formulation, and reproduced

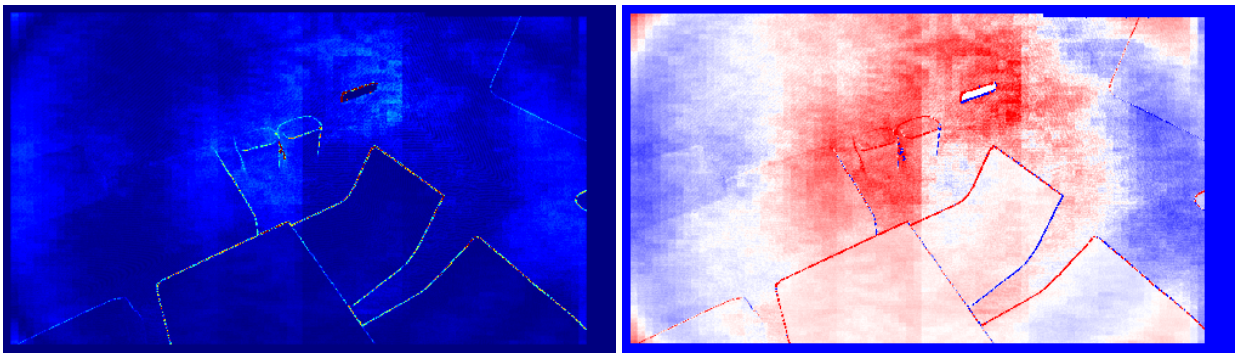


Figure 4.2: Absolute and signed depth difference between noiseless and simulated noisy depth maps.

without the log terms in Eqs. (4.3) and (4.4). Within the tracking loss, we incorporate the rendered uncertainty $\hat{S}_D^f$ and the proxy error by summing the two together. We motivate this by the idea of scaling the loss if we are either uncertain in the measurement or in the model.

$$\mathcal{L}_{g,map} = \frac{1}{M} \sum_{m=1}^{M} \frac{|D_m - \hat{D}_m|}{\beta_m} \tag{4.3}$$

$$\mathcal{L}_{g,track} = \frac{1}{M_t} \sum_{m=1}^{M_t} \frac{|D_m - \hat{D}_m|}{\beta_m + \sqrt{\hat{S}_D^f}} \tag{4.4}$$

4.4.3 3D Reconstruction using Error-scaled Loss

We investigate the mapping-only, or 3D reconstruction, performance of NICE-SLAM on the SFN-Replica dataset by employing GT poses. We present the mean and standard deviation, presented in parentheses, of the SL and SGM models in Table 4.4.

Table 4.4: 3D Reconstruction evaluation metrics comparing the original loss and the scaled proxy loss for the simulated depth maps in SFN-Replica. **Bolded** shows improvement. *Italics* shows degradation.

Scene	Sensor	Track Loss	Map Loss	N	ATE [m]	Acc. [cm]	Comp. [cm]	C. Ratio [%]	2D L1 [cm]
Office 0	SL	-	OG	15	-	3.03 (0.17)	2.08 (0.02)	94.5 (0.2)	1.80 (0.03)
Office 1	SL	-	OG	15	-	2.71 (0.23)	1.66 (0.06)	96.1 (0.4)	1.37 (0.03)
Room 2	SL	-	OG	15	-	2.80 (0.09)	2.39 (0.02)	92.6 (0.3)	2.11 (0.02)
Office 0	SL	-	Proxy	10	-	**2.82** (0.16)	**1.94** (0.02)	**94.8** (0.2)	**1.56** (0.04)
Office 1	SL	-	Proxy	10	-	**2.55** (0.27)	**1.58** (0.03)	**96.5** (0.3)	**1.26** (0.05)
Room 2	SL	-	Proxy	10	-	**2.72** (0.09)	**2.26** (0.03)	**93.2** (0.3)	**1.86** (0.03)
Office 0	SGM	-	OG	15	-	13.19 (0.83)	2.56 (0.03)	91.1 (0.3)	2.39 (0.04)
Office 1	SGM	-	OG	15	-	44.19 (3.09)	3.00 (0.04)	88.0 (0.3)	3.93 (0.11)
Room 2	SGM	-	OG	15	-	14.03 (1.72)	2.80 (0.05)	89.6 (0.5)	3.03 (0.05)
Office 0	SGM	-	Proxy	10	-	**12.35** (1.07)	**2.30** (0.04)	**92.7** (0.2)	**1.91** (0.06)
Office 1	SGM	-	Proxy	10	-	*45.84* (3.24)	**2.71** (0.03)	**89.4** (0.3)	**3.44** (0.07)
Room 2	SGM	-	Proxy	10	-	**13.12** (1.20)	**2.35** (0.04)	**94.0** (0.4)	**2.04** (0.04)

We see that the scaled loss improves 3D reconstruction metrics in nearly all cases and in all cases for the 2D rendering metrics. We note, encouragingly, that 23/24 of the mean metrics have improved using the proxy scaling method.

4.4.4 3D SLAM using Error-scaled Loss

We also investigate the effect of using scaled losses for the tracking component of NICE-SLAM as discussed in Section 4.4.4. Table 4.5 shows the tracking error, reconstruction, and rendering performance comparisons.

Table 4.5: 3D SLAM evaluation metrics comparing the original loss and the scaled proxy loss for the simulated depth maps in SFN-Replica. **Bolded** shows improvement.

Scene	Sensor	Track Loss	Map Loss	N	ATE [m]	Acc. [cm]	Comp. [cm]	C. Ratio [%]	2D L1 [cm]
Office 0	SL	OG	OG	10	0.26 (0.03)	7.90 (0.56)	7.37 (0.94)	60.4 (3.0)	12.83 (1.38)
Office 1	SL	OG	OG	10	0.52 (0.24)	12.52 (4.03)	10.26 (1.76)	42.0 (8.8)	16.01 (4.67)
Room 2	SL	OG	OG	10	0.40 (0.34)	11.90 (7.73)	8.35 (3.10)	56.0 (7.6)	17.06 (11.7)
Office 0	SL	Proxy	Proxy	10	**0.23** (0.04)	**7.00** (0.51)	**6.40** (0.69)	**63.8** (3.1)	**10.72** (1.29)
Office 1	SL	Proxy	Proxy	10	**0.33** (0.06)	**8.95** (2.51)	**8.14** (0.73)	**52.5** (5.0)	**11.07** (1.52)
Room 2	SL	Proxy	Proxy	10	**0.16** (0.05)	**4.00** (0.88)	**3.54** (0.67)	**81.1** (5.8)	**5.03** (1.32)
Office 0	SGM	OG	OG	10	0.18 (0.04)	15.22 (1.85)	5.92 (0.73)	63.2 (3.5)	9.67 (1.27)
Office 1	SGM	OG	OG	10	0.47 (0.15)	46.79 (5.21)	8.96 (1.39)	41.4 (8.2)	20.08 (8.63)
Room 2	SGM	OG	OG	10	0.22 (0.05)	15.63 (1.91)	4.61 (0.86)	72.1 (6.2)	7.24 (1.93)
Office 0	SGM	Proxy	Proxy	9	**0.17** (0.04)	**14.25** (1.23)	**5.25** (0.63)	**66.3** (4.1)	**8.06** (1.32)
Office 1	SGM	Proxy	Proxy	10	**0.36** (0.22)	**45.50** (4.80)	**6.53** (0.95)	**54.5** (7.4)	**10.57** (1.91)
Room 2	SGM	Proxy	Proxy	10	**0.20** (0.08)	**14.22** (1.57)	**3.47** (0.49)	**81.0** (5.9)	**5.51** (1.58)

The proxy-based loss achieves improvement across all metrics (30/30) compared to the original implementation of NICE-SLAM. In particular, we find that SL Room 2 performance greatly improves when using the proxy scaling. We also note that the introduction of tracking increases the variance in performance, which is shown by the ratio of significant results in the following section. Nevertheless, seeing improvements across all metrics with tracking enabled is highly encouraging for pursuing online learned uncertainty.

4.4.5 Statistical Analysis of Improvements

As discussed in Section 4.1, we also look at results using statistical methods. By employing unpaired t-tests, we can see if the improvements meet our threshold of statistical significance ($P < 0.05$). We note that from the previous sections, 53/54 metrics have shown improvement using the proxy error as a scaling term. In Table 4.6, we show the results of the unpaired t-tests when comparing the original and the newly implemented methods.

We find that 42/54 differences are statistically significant. Of these 42 significant results, we find that every metric has improved. Of the 12 results that failed to meet our criteria for statistical significance, we find that 11 results have improved. Among the mapping-only metrics, we find that 4/24 metrics do not meet our threshold for significance. Among the tracking-enabled reconstructions, we find that 8/30 metrics do not meet our threshold for significance. This overall improvement is compelling evidence that learning uncertainty can improve the NICE-SLAM's overall performance for accuracy and completion.

Table 4.6: Statistical significance of differences based on Welch's t-test. *Italicized* results are not statistically significant ($P > 0.05$). Grayed results degraded using the proxy-based loss.

		Structured Light					SGM Stereo				
Scene	Track	P ATE	P Acc.	P Comp.	P Ratio	P 2D	P ATE	P Acc.	P Comp.	P Ratio	P 2D
Office 0	-	-	0.8% ↑	0.0% ↑	0.4% ↑	0.0% ↑	-	*5.1%* ↑	0.0% ↑	0.0% ↑	0.0% ↑
Office 1	-	-	*8.1%* ↑	0.1% ↑	4.1% ↑	0.0% ↑	-	*21.9%* ↓	0.0% ↑	0.0% ↑	0.0% ↑
Room 2	-	-	1.3% ↑	0.0% ↑	0.0% ↑	0.0% ↑	-	*13.2%* ↑	0.0% ↑	0.0% ↑	0.0% ↑
Office 0	✓	4.6% ↑	0.1% ↑	1.8% ↑	2.3% ↑	0.2% ↑	*41.7%* ↑	*19.3%* ↑	4.6% ↑	*9.4%* ↑	1.5% ↑
Office 1	✓	3.3% ↑	3.1% ↑	0.4% ↑	0.5% ↑	0.9% ↑	*20.3%* ↑	*57.0%* ↑	0.0% ↑	0.1% ↑	0.7% ↑
Room 2	✓	*5.3%* ↑	1.0% ↑	0.1% ↑	0.0% ↑	1.0% ↑	*39.4%* ↑	*8.7%* ↑	0.2% ↑	0.4% ↑	4.1% ↑

4.5 Architecture Evaluation

Given the different uncertainty architectures proposed in Section 3.4, we evaluate their performance effect on SFN-Replica under the SL noise model. In these evaluations, we ensure we have five runs at a minimum to provide a sample population for our statistical tests.

We select four variables to vary in these experiments to understand which architecture has the most promising results. First we have the two different architectures for rendering uncertainty: 3D grid or 2D patch. The ray-based MLP is equivalent to the patch-based MLP with a patch size of one. Next we vary the minimum uncertainty value β_{min}: 1e-1 m or 1e-3 m. We vary the kernel size or patch size: 1×1 or 5×5. Lastly we select two options for the informative features. The first option is to use the depth and the incident angle for a pixel feature dimension of two. The second option is to use the depth, the normal direction, the image gradients, and the incident angle for a pixel feature dimension of seven. In total, we perform 16 ablations, whose trials[1] detailed in Table 4.7, determine the best performing architecture.

Table 4.7: Description of different ablations for understanding the effect of different architectural and loss methods.

Run	Architecture	Patch-size	D_m	$\mathbf{N}_m$	dx, dy	θ	β_{min}	Notes
2D1K7FS	2D Ray	1	✓	✓	✓	✓	1e-3	SFN-Replica
2D1K2FS	2D Ray	1	✓	-	-	✓	1e-3	SFN-Replica
2D1K7FL	2D Ray	1	✓	✓	✓	✓	1e-1	SFN-Replica
2D1K2FL	2D Ray	1	✓	-	-	✓	1e-1	SFN-Replica
2D5K7FS	2D Patch	5	✓	✓	✓	✓	1e-3	SFN-Replica
2D5K2FS	2D Patch	5	✓	-	-	✓	1e-3	SFN-Replica
2D5K7FL	2D Patch	5	✓	✓	✓	✓	1e-1	SFN-Replica
2D5K2FL	2D Patch	5	✓	-	-	✓	1e-1	SFN-Replica
3D1K7FS	3D Grid	1	✓	✓	✓	✓	1e-3	SFN-Replica
3D1K2FS	3D Grid	1	✓	-	-	✓	1e-3	SFN-Replica
3D1K7FL	3D Grid	1	✓	✓	✓	✓	1e-1	SFN-Replica
3D1K2FL	3D Grid	1	✓	-	-	✓	1e-1	SFN-Replica
3D5K7FS	3D Grid	5	✓	✓	✓	✓	1e-3	SFN-Replica
3D5K2FS	3D Grid	5	✓	-	-	✓	1e-3	SFN-Replica
3D5K7FL	3D Grid	5	✓	✓	✓	✓	1e-1	SFN-Replica
3D5K2FL	3D Grid	5	✓	-	-	✓	1e-1	SFN-Replica

[1] We provide trial names based on the ablation parameters. These involve the MLP architecture, the patch size, the number of features, and the use of a "small" or "large" regularizer: [2D/3D][1K/5K][2F/7F][S/L]

4.5.1 2D Network Ablations

We first present results using the 2D feature map MLP using a 1×1 patch. Within this subset, we have four ablation results between the choice of regularizer and the number of input features. These results are summarized in Table 4.8.

Table 4.8: 3D SLAM evaluation metrics comparing the uncertainty-aware losses in SFN-Replica using the 2D ray MLP architecture. **Bolded** shows improvement. *Italicized* shows degradation. "2D1K2FL" achieves the most consistent improvement across metrics.

Scene	Sensor	Track Loss	Map Loss	N	ATE [m]	Acc. [cm]	Comp. [cm]	C. Ratio [%]	2D L1 [cm]
Office 0	SL	-	2D1K7FS	10	-	**2.97** (0.19)	**2.06** (0.02)	**94.7** (0.3)	**1.79** (0.03)
Office 1	SL	-	2D1K7FS	10	-	*2.75* (0.19)	**1.64** (0.04)	**96.3** (0.3)	**1.35** (0.05)
Room 2	SL	-	2D1K7FS	10	-	*2.88* (0.12)	*2.41* (0.02)	*92.4* (0.2)	*2.14* (0.05)
Office 0	SL	-	2D1K7FL	10	-	*3.24* (0.34)	**2.07** (0.03)	**94.7** (0.3)	*2.73* (1.44)
Office 1	SL	-	2D1K7FL	10	-	*2.80* (0.21)	**1.65** (0.03)	96.2 (0.3)	**1.36** (0.03)
Room 2	SL	-	2D1K7FL	9	-	*2.81* (0.06)	2.39 (0.03)	*92.5* (0.4)	*2.12* (0.03)
Office 0	SL	-	2D1K2FS	10	-	**2.93** (0.13)	*2.10* (0.02)	*94.4* (0.2)	*1.90* (0.03)
Office 1	SL	-	2D1K2FS	10	-	**2.63** (0.20)	**1.62** (0.03)	**96.4** (0.3)	*1.39* (0.05)
Room 2	SL	-	2D1K2FS	10	-	*2.91* (0.14)	*2.49* (0.02)	*91.7* (0.4)	*2.22* (0.02)
Office 0	SL	-	2D1K2FL	10	-	**2.85** (0.09)	**2.05** (0.02)	**94.8** (0.2)	1.80 (0.03)
Office 1	SL	-	2D1K2FL	10	-	*2.80* (0.15)	**1.61** (0.03)	**96.5** (0.3)	*1.40* (0.10)
Room 2	SL	-	2D1K2FL	10	-	**2.76** (0.11)	**2.38** (0.03)	92.6 (0.3)	**2.09** (0.04)

Using the 1×1 patch-based approach, we find improvement over some parameters and degradation in others. We find that the simplest architecture "2D1K2FL", employing two input features and a larger β_{min}, has one of the better performances within this subset of ablations. This method improves across eight metrics and observes degradation in two metrics. The remaining two metrics are within rounding error. With few input parameters and a larger regularizer, the chance of overfitting may be limited by this particular architecture, preventing the more wide-spread degradation we observe across other trials.

We next present results using the 2D feature map MLP using a 5×5 patch. Within this subset, we have four ablation results between the use of regularizer and the number of input features. These results are summarized in Table 4.9.

Within the patch-base approach, we find two methods achieve positive improvement across a majority of metrics. Trials "2D5K7FL" and "2D5K2FS" both see improvements across eight metrics. "2D5K2FS" saw fewer metrics degrade in performance after discounting rounding errors. Overall, however, both these methods achieve marginal improvement over the baseline methods. The "2D5K7FS" trial experienced a strong outlier that skewed results in the Room 2 scenes, as seen by how much its metrics diverge from the other results. The other three trials congregate around a similar performance cluster. We see again that the use of a larger regularizer β_{min} may be beneficial within the single sensor framework in improving metrics. When using a smaller regularizer, the inclusion of fewer features may improve results.

However, the data we have is inconclusive and the evidence is limited in the above claims. We see that that "2D5K2FL" appears to perform worse than "2D5K2FS," which contradicts our belief that strong regularizers should be beneficial in 3D reconstruction. Amongst the 2D MLP approaches, we decide to use "2D5K2FS" as the architecture of choice for further ablations.

Table 4.9: 3D SLAM evaluation metrics comparing the uncertainty-aware losses in SFN-Replica using the 2D patch MLP architecture. **Bolded** shows improvement. *Italicized* shows degradation. "2D5K2FS" achieves the most consistent improvement across metrics.

Scene	Sensor	Track Loss	Map Loss	N	ATE [m]	Acc. [cm]	Comp. [cm]	C. Ratio [%]	2D L1 [cm]
Office 0	SL	-	2D5K7FS	10	-	*3.06* (0.09)	2.08 (0.03)	94.5 (0.4)	*1.86* (0.02)
Office 1	SL	-	2D5K7FS	10	-	*2.72* (0.27)	**1.62** (0.03)	**96.4** (0.3)	*1.39* (0.06)
Room 2	SL	-	2D5K7FS	10	-	*2.87* (0.17)	*2.45* (0.01)	*91.9* (0.2)	*2.56* (1.25)
Office 0	SL	-	2D5K7FL	10	-	*3.08* (0.18)	**2.06** (0.03)	**94.7** (0.3)	*1.82* (0.04)
Office 1	SL	-	2D5K7FL	10	-	**2.68** (0.20)	**1.61** (0.02)	**96.6** (0.2)	**1.36** (0.03)
Room 2	SL	-	2D5K7FL	7	-	*2.85* (0.06)	**2.37** (0.04)	**92.8** (0.4)	2.11 (0.01)
Office 0	SL	-	2D5K2FS	10	-	**2.87** (0.09)	**2.06** (0.04)	**94.7** (0.3)	**1.79** (0.02)
Office 1	SL	-	2D5K2FS	10	-	*2.75* (0.21)	**1.61** (0.03)	**96.5** (0.3)	**1.36** (0.04)
Room 2	SL	-	2D5K2FS	10	-	**2.79** (0.15)	2.39 (0.02)	92.6 (0.3)	2.11 (0.05)
Office 0	SL	-	2D5K2FL	10	-	3.03 (0.19)	**2.07** (0.02)	**94.6** (0.2)	1.80 (0.03)
Office 1	SL	-	2D5K2FL	10	-	*2.84* (0.19)	**1.60** (0.03)	**96.7** (0.3)	*1.39* (0.04)
Room 2	SL	-	2D5K2FL	10	-	*2.81* (0.14)	*2.40* (0.03)	*92.4* (0.4)	2.11 (0.02)

4.5.2 3D Network Ablations

Lastly, we present results using the 3D feature grid-based MLP using appended features as described in Section 3.4.1. This approach leverages volume rendering of the uncertainty and allows for the accumulation of uncertainty across frames. The use of local 2D features can additionally provide direct information of the individual frames and the specific noise associated from the sensor or depth generating method. Within this subset, we have eight ablation results varying the patch size, the choice of regularizer, and the number of input features. These results are summarized in Table 4.10.

We find some broad trends within this set of ablations. The use of a small regularizer has a clear detrimental effect on some of the 3D reconstruction metrics, with obvious degradation in the 2D rendering metrics and within the completion ratio criteria for Room 2. Room 2 in particular appears challenging when using a small regularizer. The remaining four methods have similar performance overall, but we note that trials "3D5K2FL" and "3D1K2FL" appear to have better performance, seeing improvement in 10/12 and 11/12 metrics in total. These two methods only include the depth and the incident angle as appended features, with one using a 5×5 patch and the other using a 1×1 patch. The inclusion of derivative and normal information does not appear to provide useful information as the 3D reconstruction metrics and 2D rendering metrics are not noticeable improved compared to the architectures using only depth and incident angle information..

Ultimately, we find that the results of using these different architectures to learn an implicit uncertainty from our constructed uncertainty-aware loss to be marginally effective. We further analyze the performance differences and the associated statistical significance in Appendix C. We find that we can see statistically significant gains in performance using our learned uncertainty approaches and that many of degradations we observe are not statistically significant. Following this analysis, we select the "3D1K2FL" architecture as our representative 3D architecture of choice.

Table 4.10: 3D SLAM evaluation metrics comparing the uncertainty-aware losses for the simulated depth maps in SFN-Replica using the 3D feature grid architecture. **Bolded** shows improvement. *Italicized* shows degradation. "3D1K2FL" achieves the most consistent improvement across metrics. "3D1K2FL" achieves the most consistent improvement across metrics.

Scene	Sensor	Track Loss	Map Loss	N	ATE [m]	Acc. [cm]	Comp. [cm]	C. Ratio [%]	2D L1 [cm]
Office 0	SL	-	3D1K7FS	10	-	*3.04* (0.13)	*2.10* (0.01)	94.5 (0.2)	*1.86* (0.03)
Office 1	SL	-	3D1K7FS	10	-	*2.83* (0.17)	**1.62** (0.03)	**96.4** (0.3)	**1.34** (0.06)
Room 2	SL	-	3D1K7FS	9	-	*2.84* (0.10)	*2.44* (0.03)	*92.1* (0.3)	*2.16* (0.03)
Office 0	SL	-	3D1K7FL	6	-	**2.89** (0.16)	*2.09* (0.02)	*94.4* (0.2)	1.80 (0.02)
Office 1	SL	-	3D1K7FL	9	-	*2.91* (0.24)	**1.61** (0.02)	**96.5** (0.2)	**1.36** (0.04)
Room 2	SL	-	3D1K7FL	9	-	**2.73** (0.10)	**2.38** (0.03)	92.6 (0.2)	**2.08** (0.03)
Office 0	SL	-	3D1K2FS	10	-	**2.85** (0.14)	*2.09* (0.03)	**94.6** (0.3)	*1.87* (0.03)
Office 1	SL	-	3D1K2FS	10	-	*2.74* (0.20)	**1.63** (0.04)	**96.3** (0.3)	*1.40* (0.07)
Room 2	SL	-	3D1K2FS	7	-	*2.94* (0.14)	*2.47* (0.03)	*91.9* (0.3)	*2.17* (0.04)
Office 0	SL	-	3D1K2FL	10	-	**2.90** (0.14)	**2.05** (0.02)	**94.8** (0.2)	**1.78** (0.02)
Office 1	SL	-	3D1K2FL	10	-	**2.68** (0.15)	**1.59** (0.02)	**96.7** (0.2)	*1.38* (0.03)
Room 2	SL	-	3D1K2FL	8	-	**2.77** (0.11)	**2.37** (0.03)	**92.8** (0.4)	**2.08** (0.03)
Office 0	SL	-	3D5K7FS	10	-	**2.95** (0.18)	*2.11* (0.02)	*94.4* (0.2)	*1.87* (0.02)
Office 1	SL	-	3D5K7FS	7	-	**2.59** (0.13)	**1.61** (0.03)	**96.4** (0.3)	**1.33** (0.04)
Room 2	SL	-	3D5K7FS	10	-	**2.74** (0.09)	*2.44* (0.02)	*92.0* (0.2)	*2.17* (0.04)
Office 0	SL	-	3D5K7FL	10	-	**2.88** (0.19)	**2.06** (0.02)	**94.7** (0.2)	**1.77** (0.02)
Office 1	SL	-	3D5K7FL	10	-	*2.75* (0.25)	**1.60** (0.03)	**96.5** (0.2)	**1.35** (0.04)
Room 2	SL	-	3D5K7FL	10	-	**2.74** (0.09)	**2.38** (0.02)	92.6 (0.2)	**2.10** (0.03)
Office 0	SL	-	3D5K2FS	8	-	**2.88** (0.15)	*2.11* (0.03)	94.5 (0.1)	*1.89* (0.03)
Office 1	SL	-	3D5K2FS	6	-	**2.68** (0.24)	**1.62** (0.03)	**96.4** (0.3)	**1.31** (0.08)
Room 2	SL	-	3D5K2FS	5	-	**2.76** (0.14)	*2.45* (0.02)	*91.9* (0.3)	*2.18* (0.02)
Office 0	SL	-	3D5K2FL	6	-	**2.84** (0.18)	2.08 (0.02)	**94.6** (0.2)	**1.79** (0.02)
Office 1	SL	-	3D5K2FL	6	-	*2.78* (0.09)	**1.61** (0.02)	**96.6** (0.2)	**1.34** (0.02)
Room 2	SL	-	3D5K2FL	6	-	**2.67** (0.07)	**2.37** (0.04)	**92.7** (0.3)	**2.10** (0.03)

4.5.3 Ablation Visualization

To better visualize and draw conclusions from the ablations above, we plot the performance metrics against one another for the different trials, including the original and proxy-based approach. We compare accuracy and completion, which capture complimentary information in the scene reconstruction. We highlight the proxy-based approach in green and the original approach in red. These graphs are presented in Figure 4.3.

We can clearly see that our constructed proxy-based loss outperforms all other methods by a significant margin. Some obvious trends are also visible in these plots—*e.g.* we can see that the 3D grid approaches perform better with the larger regularizer (brown) over the weaker regularizer (orange). We can also see scene dependence on the performance of using our implicitly learned uncertainty. For example in "Office 0," we typically observe improvement to accuracy, but less method-agnostic effects on completion. In "Office 1," we find that most approaches improve on completion, but have inconclusive effects on accuracy. In 'Room 2," we see the clearest clustering by the choice of base architecture and the choice of regularizer. The methods with larger $\beta_{\min}$ cluster more closely to the original loss performance without degradation.

We also compare the completion ratio and the depth rendering loss in Figure 4.4. We see a clustering of results here among different regularizer and between the 2D vs. 3D methods. Typically depth L1 rendering

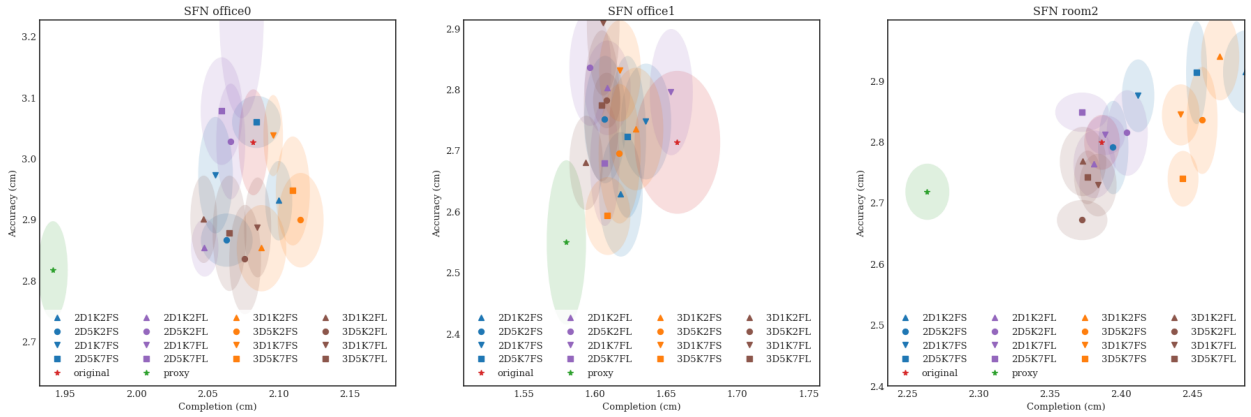


Figure 4.3: Comparison plots of accuracy and completion for the original, proxy-based, and learned uncertainty methods. The ellipses in lower opacity represents the standard deviation. Lower left is better.

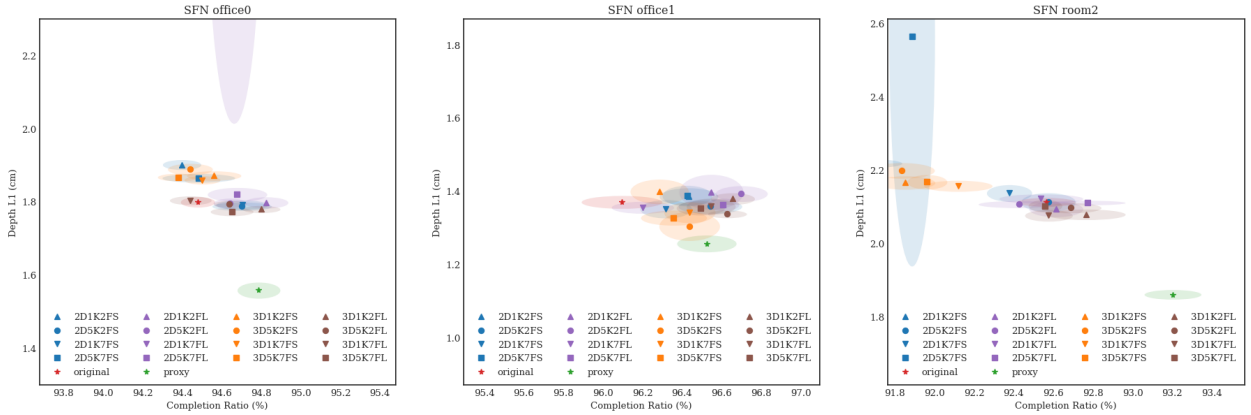


Figure 4.4: Comparison plots of completion ratio and depth L1 for the original, proxy-based, and learned uncertainty methods. The ellipses in lower opacity represents the standard deviation. Lower right is better.

is not improved, but we can see improvement in completion ratio for certain methods.

Overall, we find that just from the information provided from a single sensor, we fail to attain the same kind of performance as seen in our constructed proxy-based loss. Our proxy-based loss sets a loosely theoretical "upper bound" on the performance gain we may observe from learning uncertainty.

4.6 Loss Function Ablation

We introduced two modifications in Section 3.3 that ensure we would not encounter the zero uncertainty or infinite confidence condition. Without these modifications, we find that the code fails from instability due to divisions by zero.

Among the two modifications, we perform cursory explorations using the NS-Replica dataset with the SL noise model. We utilize the "2D5K2F" architecture sing 2D feature map information, a patch size of five, and the two image features: depth and incident angle. For the confidence based network, we set $\lambda_c = 0.015$ as was used in [44]. For the uncertainty architectures, we set the regularizer as $\beta_{\min} = 0.001$. We summarize the results in Table 4.11. We use "2D5K2FC" as the key for the modified approach using confidence.

Table 4.11: 3D Reconstruction evaluation metrics comparing the original loss to the confidence and uncertainty-aware loss in SFN-Replica. **Bolded** shows improvement. *Italics* shows degradation. No statistically significant differences were found between confidence and uncertainty-modified approaches.

Scene	Sensor	Track Loss	Map Loss	N	ATE [m]	Acc. [cm]	Comp. [cm]	C. Ratio [%]	2D L1 [cm]
Office 0	SL	-	OG	10	-	2.89 (0.20)	9.92 (0.33)	82.0 (0.2)	3.23 (0.06)
Office 1	SL	-	OG	10	-	3.36 (0.38)	8.51 (0.10)	81.4 (0.3)	11.77 (0.08)
Room 2	SL	-	OG	10	-	2.46 (0.04)	4.26 (0.07)	87.2 (0.3)	7.69 (0.08)
Office 0	SL	-	2D5K2FC	6	-	**2.75** (0.17)	**9.81** (0.49)	*81.9* (0.2)	*3.26* (0.14)
Office 1	SL	-	2D5K2FC	6	-	**3.28** (0.35)	*8.58* (0.09)	*81.2* (0.3)	*11.79* (0.14)
Room 2	SL	-	2D5K2FC	6	-	**2.43** (0.06)	*4.27* (0.09)	*87.1* (0.4)	*7.70* (0.10)
Office 0	SL	-	2D5K2FS	6	-	**2.80** (0.29)	**9.83** (0.50)	*81.9* (0.2)	*3.26* (0.13)
Office 1	SL	-	2D5K2FS	6	-	3.36 (0.24)	*8.56* (0.21)	81.4 (0.4)	**11.75** (0.10)
Room 2	SL	-	2D5K2FS	6	-	**2.44** (0.03)	*4.27* (0.03)	**87.3** (0.2)	**7.62** (0.07)

We fail to find statistically significant differences resulting from these two methods. While the confidence-based formulating might have desirable properties for bounding the loss function and controlling the uncertainty scaling via λ_c, we decide to proceed using the uncertainty approach over the confidence approach.

4.7 Dataset Exploration & Generalization

4.7.1 SFN-Replica

Our ablations have focused on 3D reconstruction and have not explored the SLAM setting for leveraging uncertainty. We perform experiments using the SFN-Replica dataset with tracking enabled to see if we can achieve the same performance gain as in our proxy-based approach described previously. We perform experiments using the "2D5K2FS" and "3D1K2FL" architectures, which show some of the better performances from our architecture ablations. Table 4.12 shows the full comparison of metrics between the original, proxy-based, and learned methods.

Table 4.12: 3D SLAM evaluation metrics comparing the original, proxy-based, and the uncertainty-aware loss for the simulated depth maps in SFN-Replica. **Bolded** shows improvement. More consistent improvement across metrics was achieve using the "2D5K2FS" architecture.

Scene	Sensor	Track Loss	Map Loss	N	ATE [m]	Acc. [cm]	Comp. [cm]	C. Ratio [%]	2D L1 [cm]
Office 0	SL	OG	OG	10	0.26 (0.03)	7.90 (0.56)	7.37 (0.94)	60.4 (3.0)	12.83 (1.38)
Office 1	SL	OG	OG	10	0.52 (0.24)	12.52 (4.03)	10.26 (1.76)	42.0 (8.8)	16.01 (4.67)
Room 2	SL	OG	OG	10	0.40 (0.34)	11.90 (7.73)	8.35 (3.10)	56.0 (7.6)	17.06 (11.7)
Office 0	SL	Proxy	Proxy	10	**0.23** (0.04)	**7.00** (0.51)	**6.40** (0.69)	**63.8** (3.1)	**10.72** (1.29)
Office 1	SL	Proxy	Proxy	10	**0.33** (0.06)	**8.95** (2.51)	**8.14** (0.73)	**52.5** (5.0)	**11.07** (1.52)
Room 2	SL	Proxy	Proxy	10	**0.16** (0.05)	**4.00** (0.88)	**3.54** (0.67)	**81.1** (5.8)	**5.03** (1.32)
Office 0	SL	2D5K2FS		10	*0.27* (0.03)	**7.46** (0.866	*7.57* (0.91)	**62.3** (2.0)	**11.23** (1.28)
Office 1	SL	2D5K2FS		10	**0.35** (0.11)	**9.35** (3.12)	**8.23** (1.46)	**51.6** (6.5)	**10.92** (3.71)
Room 2	SL	2D5K2FS		10	**0.32** (0.06)	**8.49** (2.41)	**7.72** (2.19)	**59.3** (6.8)	**13.96** (4.72)
Office 0	SL	3D1K2FL		10	**0.25** (0.04)	**7.73** (0.90)	**6.91** (0.72)	*60.2* (4.1)	**11.80** (1.50)
Office 1	SL	3D1K2FL		10	**0.49** (0.22)	*13.46* (5.80)	*10.96* (2.37)	*40.1* (9.5)	*17.42* (9.44)
Room 2	SL	3D1K2FL		10	**0.30** (0.08)	**10.64** (3.85)	**8.07** (3.22)	**60.2** (11.5)	**14.64** (6.96)

Overall, we find that the "'2D5K2FS" method improves across 13/15 metrics, while "3D1K2FL" improves across 10/15 metrics compared to the original implementation. We also find that of the improvements using the "2D5K2FS" method, four meet our threshold for significance and the two methods which degrade are not found to meet our threshold. No result, improvement or degradation, in the "3D1K2FL" method was found to have statistical significance.

Overall, we find that the improvement we can observe on the SFN-Replica dataset using our implicitly-learned uncertainty to be highly encouraging. The results using solely the per-frame information improving across the majority of metrics is a particularly motivating result. The 3D volume-rendered approach has the capacity to accumulate uncertainty information across the trajectory, but adds a pose-dependency that becomes an issue when tracking is enabled.

Furthermore, we find that the 2D frame-only approach is able to achieve more convincing qualitative results. We showcase these rendered uncertainties in Figure 4.5. In these rendered uncertainties, the 3D approach returns uncertainty maps that appear uninformative. We especially note the rendered uncertainty for Office 1, where we see a very homogeneously rendered uncertainty map. In Office 0 and Room 2, the 3D approach captures similar areas of high uncertainty as the ground-truth proxy error, but in a low-fidelity fashion. The 2D approach, in contrast, exhibits similar distributions with edge and depth dependencies as the ground-truth proxy error.

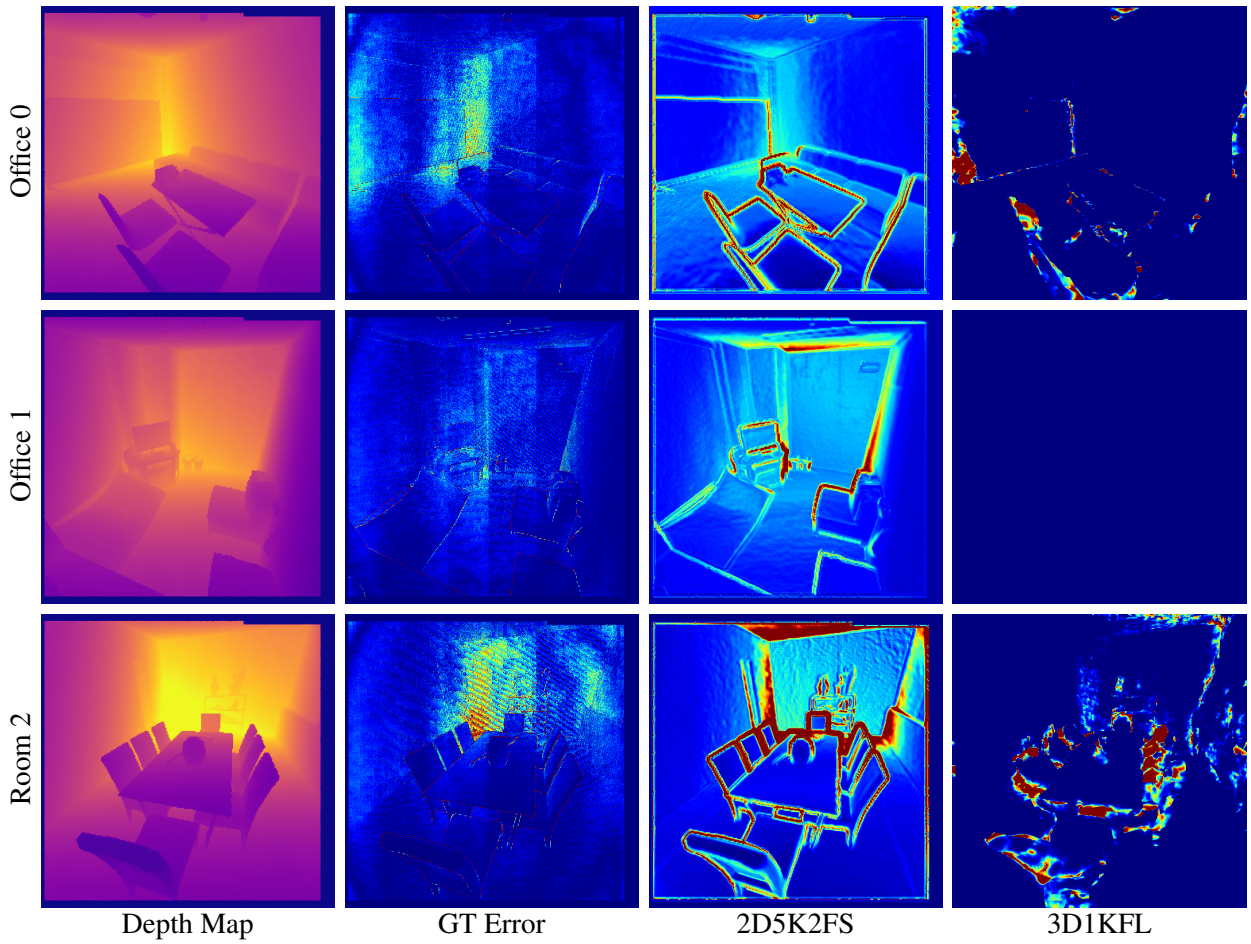


Figure 4.5: Comparison of depth maps, generated uncertainty, and the ground-truth proxy error for the results with tracking enabled on the SFN-Replica dataset.

4.7.2 NS-Replica

The SFN-Replica dataset provides challenging trajectories that are coarse in nature. Typical streams from real-world devices tend to have smoother trajectories. Fortunately, the NS-Replica trajectories, originally developed in iMAP [41], exhibit more realistic trajectories. These datasets, however, do not observe the entire scene. In NICE-SLAM's original evaluation, unseen portions of the mesh are culled to isolate the metrics to only observed portions of the scene. We evaluate without this culling process in place. One could argue that such an approach tests the methods ability for geometric extrapolation. We test this dataset using both tracking-enabled and mapping-only conditions. We present the results in Table 4.13.

Overall, the results are less encouraging than the SFN-Replica dataset. We find degradation across metrics for both the tracking-enabled and mapping-only conditions, and regardless of uncertainty implementation. We find that in the mapping-only process, we see improvements in one and four metrics in the "2D5K2FS" and "3D1K2FL" approaches respectively. Overall the impact is quite minor in terms of the improvement and degradation of the results and many results do not meet our threshold for significance.

Under the mapping-only conditions, one degradation is found to be statistically significant for both the 2D and 3D approach (Room 2 completion ratio). Under the tracking only conditions, the 3D approach sees

Table 4.13: 3D SLAM evaluation metrics comparing the original, proxy-based, and the uncertainty-aware loss for the simulated depth maps in NS-Replica. **Bolded** shows improvement. The improvements and degradations appear inconclusive on the smoother trajectory and higher resolution NS-Replica depth maps.

Scene	Sensor	Track Loss	Map Loss	N	ATE [m]	Acc. [cm]	Comp. [cm]	C. Ratio [%]	2D L1 [cm]
Office 0	SL	-	OG	9	-	2.89 (0.20)	9.92 (0.33)	82.0 (0.2)	3.23 (0.07)
Office 1	SL	-	OG	9	-	3.36 (0.38)	8.51 (0.10)	81.4 (0.3)	11.77 (0.08)
Room 2	SL	-	OG	10	-	2.46 (0.04)	4.26 (0.07)	87.3 (0.3)	7.69 (0.08)
Office 0	SL	-	2D5K2FS	7	-	*2.91* (0.11)	*10.05* (0.06)	*81.8* (0.2)	*3.27* (0.02)
Office 1	SL	-	2D5K2FS	11	-	*3.42* (0.23)	*8.52* (0.09)	81.4 (0.4)	*11.81* (0.13)
Room 2	SL	-	2D5K2FS	11	-	**2.42** (0.07)	*4.32* (0.08)	*87.0* (0.2)	*7.72* (0.15)
Office 0	SL	-	3D1K2FL	14	-	*3.01* (0.22)	**10.03** (0.07)	*81.8* (0.2)	3.23 (0.03)
Office 1	SL	-	3D1K2FL	9	-	**3.33** (0.44)	*8.54* (0.08)	81.4 (0.3)	*11.84* (0.09)
Room 2	SL	-	3D1K2FL	20	-	**2.44** (0.04)	**4.31** (0.11)	*87.0* (0.4)	**7.68** (0.08)
Office 0	SL	OG	OG	9	0.23 (0.04)	9.12 (2.48)	13.21 (1.26)	47.5 (4.4)	10.75 (1.90)
Office 1	SL	OG	OG	9	0.09 (0.01)	4.10 (0.32)	9.27 (0.19)	66.9 (1.7)	13.99 (0.46)
Room 2	SL	OG	OG	10	0.11 (0.01)	4.81 (0.20)	6.33 (0.22)	67.3 (1.4)	11.60 (0.38)
Office 0	SL	2D5K2FS	2D5K2FS	10	**0.22** (0.03)	**8.41** (2.18)	*13.41* (1.54)	*47.4* (4.8)	**10.69** (1.63)
Office 1	SL	2D5K2FS	2D5K2FS	10	*0.10* (0.01)	*4.42* (0.19)	*9.53* (0.24)	*64.7* (1.3)	*14.28* (0.34)
Room 2	SL	2D5K2FS	2D5K2FS	11	0.11 (0.01)	**4.28** (0.19)	**5.77** (0.23)	**70.1** (1.4)	**10.80** (0.36)
Office 0	SL	3D1K2FL	3D1K2FL	10	**0.21** (0.02)	**7.65** (1.79)	**12.51** (1.01)	**50.0** (4.6)	**9.65** (1.33)
Office 1	SL	3D1K2FL	3D1K2FL	10	*0.10* (0.01)	*4.43* (0.36)	**9.26** (0.21)	*66.4* (1.8)	*14.11* (0.34)
Room 2	SL	3D1K2FL	3D1K2FL	8	0.11 (0.01)	**4.72** (0.16)	**6.14** (0.17)	**67.7** (2.1)	**11.56** (0.51)

one statistically significant degradation (Office 1 accuracy). For the 2D approach, there are four statistically significant degradations (Office 1 ATE, accuracy, completion, and completion ratio) and four statistically significant improvements (Room 2 accuracy, completion, completion ratio, and depth L1). We present qualitative results for the mapping-only process in Figure 4.6.

We see similar trends regarding the rendering of uncertainty compared to the SFN-Replica dataset. The 2D approach using only the active frame information better reflects what we would expect from sources of measurement uncertainty, capturing depth and edge effects. The 3D approach, in contrast, looks significantly less informative and much more uniform.

The NS-Replica dataset has much smoother trajectories and a high image resolution with the same horizontal field-of-view. These factors could explain why we see a limited effect when using our uncertainty-aware approach. With a high resolution, edges are less likely to be directly sampled. When these points of high error are sampled, they will be closer to the edge itself due to the higher resolution used in the capture of this dataset. These points that may otherwise skew the tracking or mapping are less likely to be sampled, and the trajectories themselves are much easier to track.

4.7.3 TUM RGB-D

This work has focused so far on synthetic datasets for evaluation. We present results from the TUM RGB-D dataset to show the effect of our uncertainty extension on a real-world dataset. As the TUM-RGBD dataset is a real world dataset, we cannot generate 3D reconstruction metrics or the 2D depth rendering metrics since we have no access to ground-truth data. We elect to sue the 2D approach to render uncertainty due to its better qualitative performance. We can only evaluate using the ATE metric against the trajectory. For

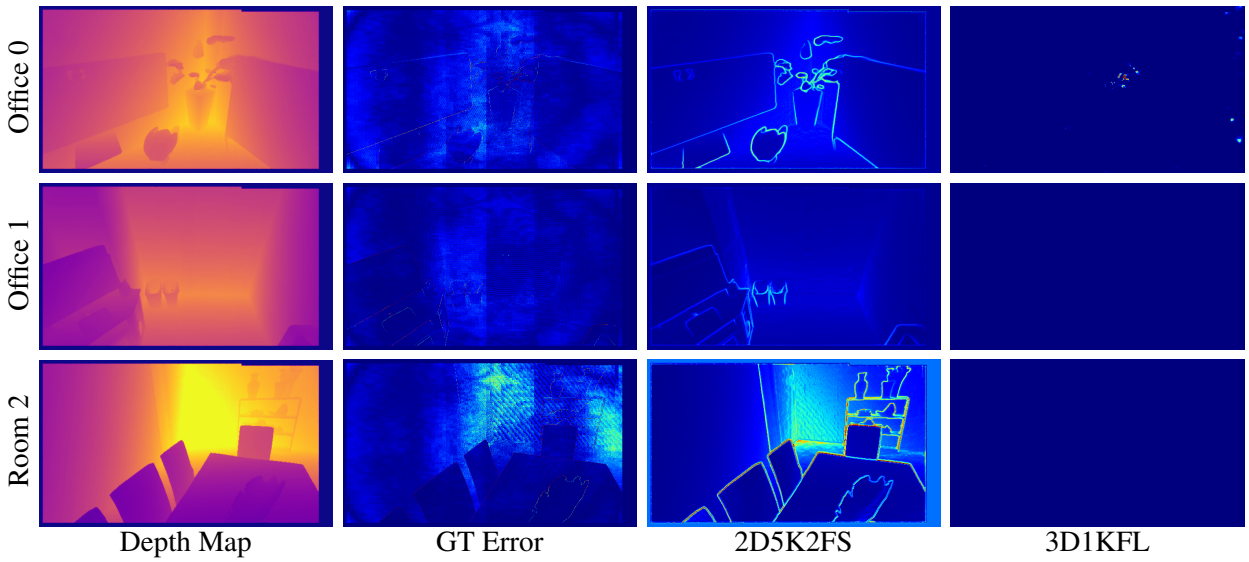


Figure 4.6: Comparison of depth maps, generated uncertainty, and the ground-truth proxy error for the results without tracking on the NS-Replica dataset.

this evaluation of ATE, we revert to typical definition of the ATE metric where we take the error after least-square minimization between the provided trajectory and our tracked trajectory. We present the visualized results of the learned uncertainty and the tracking error plots in Figure 4.7.

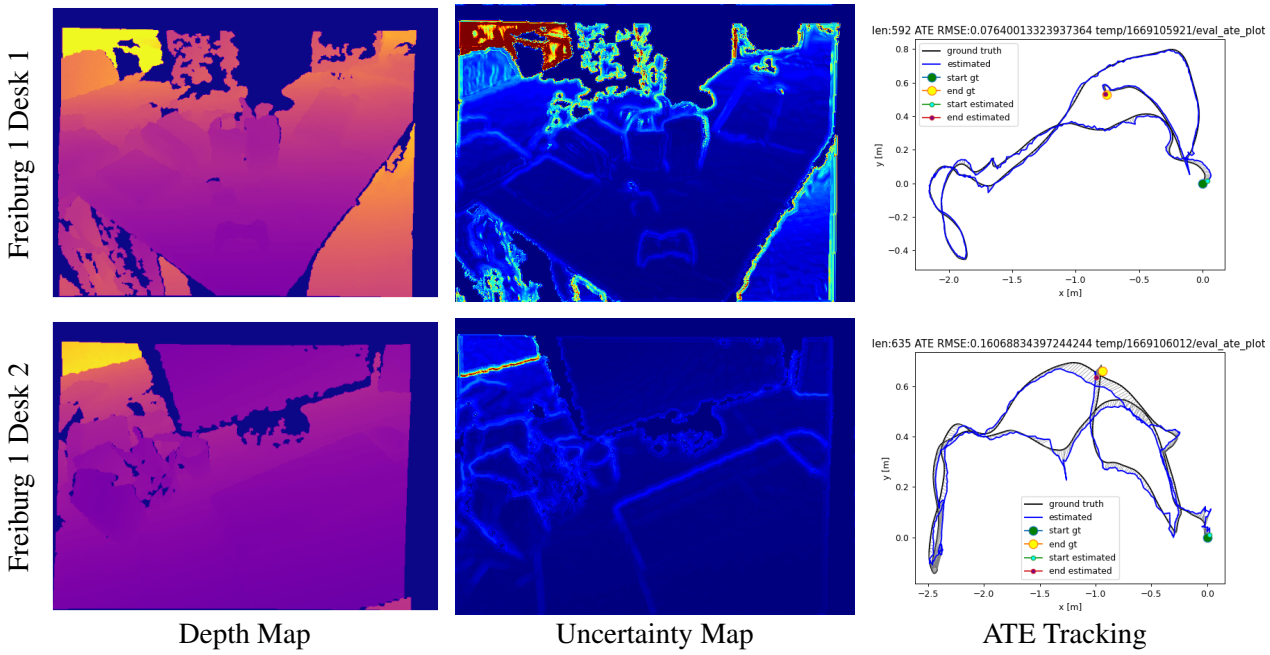


Figure 4.7: Visualization of depth maps, generated uncertainty, and the ATE tracking error results on the TUM-RGBD dataset.

The generated uncertainty is also reflective of our expectations, with clear edge and depth dependency. The ability to generate reasonable uncertainties from a real world dataset is encouraging as it signals that the work performed so far may transfer from simulation to real world applications. We also find that the trajectory tracking for both of the "Desk" scenes is generally successful. We summarize the ATE metric in Table 4.14.

Table 4.14: 3D Reconstruction evaluation metrics comparing the original loss to the uncertainty-aware loss for the TUM-RGBD dataset. **Bolded** shows improvement. *Italics* shows degradation.

Scene	Sensor	Track Loss	Map Loss	N	ATE [m]
Freiburg 1 Desk 1	SL	OG	OG	8	0.0736 (0.0084)
Freiburg 1 Desk 2	SL	OG	OG	8	0.3137 (0.2854)
Freiburg 1 Desk 1	SL	2D5K2FS	2D5K2FS	6	*0.0745* (0.0067)
Freiburg 1 Desk 2	SL	2D5K2FS	2D5K2FS	10	**0.2864** (0.3877)

With the uncertainty-aware, we find that the tracking performance is not substantially changed between using the original or the uncertainty-aware loss. We find no statistically significant difference between using the original or uncertainty-aware loss. Throughout all of the experiments, we find limited effectiveness of improving tracking results using the uncertainty-aware loss under the single-sensor environment. However, the extraction of reasonable uncertainty maps encourages us to further investigate the ability to fuse multiple sensors for improving reconstruction, tracking, and rendering.

4.8 Aligned Two-Sensor Extension

We perform experiments fusing multiple sensor observations weighted by learned uncertainty in our nerual implicit framework. We utilize the "2D5K2FS" architecture described in the previous sections as the 2D architecture has a lighter memory and computational footprint compared to the 3D grid approaches. We present in Table 4.15 the metrics using the original NICE-SLAM implementation for the SL model, SGM, and PSMNet generated depth maps, and their combinations. When performing meshing, we utilize the depth map source that attains the better reconstruction metrics[2] for determining meshing bounds to provide a fair comparison.

Table 4.15: 3D SLAM evaluation metrics comparing multi-sensor uncertainty-aware loss for the simulated depth maps in SFN-Replica. **Bolded** shows improvement. *Italicized* shows degradation. The fusion of different sensor sources generally improves 3D reconstruction metrics and more significantly improves depth rendering.

Scene	Sensor	Tr. Loss	Map Loss	N	ATE [m]	Acc. [cm]	Comp. [cm]	C. Ratio [%]	2D L1 [cm]
Office 0	SL	-	OG	15	-	3.03 (0.17)	2.08 (0.02)	94.5 (0.2)	1.80 (0.03)
Office 1	SL	-	OG	15	-	2.71 (0.23)	1.66 (0.06)	96.1 (0.4)	1.37 (0.03)
Room 2	SL	-	OG	15	-	2.80 (0.09)	2.39 (0.02)	92.6 (0.3)	2.11 (0.02)
Office 0	SGM	-	OG	15	-	13.19 (0.83)	2.56 (0.03)	91.1 (0.3)	2.39 (0.04)
Office 1	SGM	-	OG	15	-	44.19 (3.09)	3.00 (0.04)	88.0 (0.3)	3.93 (0.11)
Room 2	SGM	-	OG	15	-	14.03 (1.72)	2.80 (0.05)	89.6 (0.5)	3.03 (0.05)
Office 0	PSM	-	OG	10	-	3.27 (0.14)	2.57 (0.03)	89.9 (0.2)	2.07 (0.03)
Office 1	PSM	-	OG	10	-	4.14 (0.32)	2.58 (0.02)	87.9 (0.3)	3.29 (0.39)
Room 2	PSM	-	OG	10	-	3.95 (0.14)	2.66 (0.03)	89.3 (0.2)	3.29 (0.39)
Office 0	SL/SGM	-	2D5K2FS	12	-	**2.93** (0.18)	**2.04** (0.03)	*94.3* (0.3)	**1.74** (0.02)
Office 1	SL/SGM	-	2D5K2FS	12	-	*2.78* (0.26)	*1.68* (0.03)	*95.8* (0.2)	*1.38* (0.06)
Room 2	SL/SGM	-	2D5K2FS	12	-	*2.87* (0.12)	**2.38** (0.02)	**92.8** (0.3)	**2.06** (0.05)
Office 0	SL/PSM	-	2D5K2FS	10	-	**2.67** (0.13)	*2.10* (0.03)	*93.0* (0.2)	**1.65** (0.03)
Office 1	SL/PSM	-	2D5K2FS	10	-	*2.81* (0.17)	*1.80* (0.04)	*94.2* (0.4)	**1.35** (0.08)
Room 2	SL/PSM	-	2D5K2FS	10	-	**2.71** (0.15)	**2.21** (0.04)	**93.6** (0.5)	**1.84** (0.04)
Office 0	PSM/SGM	-	2D5K2FS	10	-	*3.54* (0.20)	**2.39** (0.03)	**91.3** (0.2)	**1.90** (0.03)
Office 1	PSM/SGM	-	2D5K2FS	10	-	**3.76** (0.23)	**2.29** (0.05)	**89.7** (0.5)	**2.18** (0.06)
Room 2	PSM/SGM	-	2D5K2FS	8	-	**3.72** (0.17)	**2.46** (0.04)	**91.3** (0.3)	**2.13** (0.03)

We find significant improvement across various metrics when using the two-sensor weighted loss function. At worst, metrics have marginally degraded compared to their best performing constituent sensor. Of the 36 metrics, we find that 24/36 metrics have improved.

While we do not see all-around improvement across metrics, we do find encouraging signs for this cursory exploration into sensor fusion with our neural implicit framework. One significant observed improvement is in the depth rendering, or 2D L1 metric, where we can see significant improvement compared to the baseline results in all scenes except for Office 1 when fusing the SL noise model and the SGM stereo method.

When fusing the SL model and SGM method, we find the performance mirrors the SL model quite closely in spite of the SGM method's worse performance overall. This is encouraging as the model learns to rely on the more trustworthy sensor. The performance degradations observed are also minor and are quite

[2] SL for SL/PSMNet and SL/SGM; PSMNet for PSMNet/SGM

similar to the base SL model results. The combination of the SL model and the PSMNet rendering also achieves the best results for 2D rendering, surpassing performance from their single-sensor counterparts. When we fuse both stereo methods, PSMNet and SGM, we find positive results across nearly all metrics. As both SGM and PSMNet have more similar performance in terms of completion, completion ratio, and depth rendering, the potential for improvement may be easier. When fusing with the SL model, the SL model exhibits better performance and limit the potential gains when fusing poorer performing sensors. We again show the comparison between the original method and the two-senor fused approaches visually in Figure 4.8.

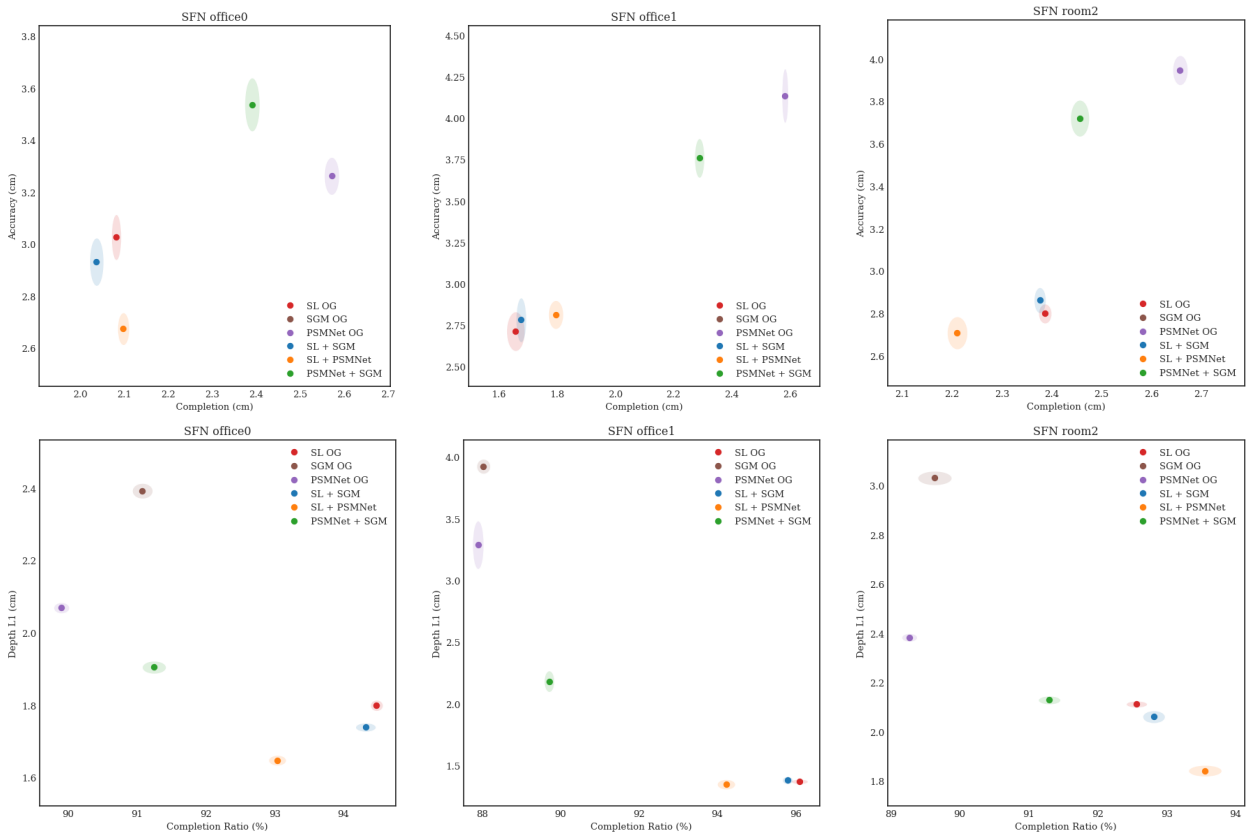
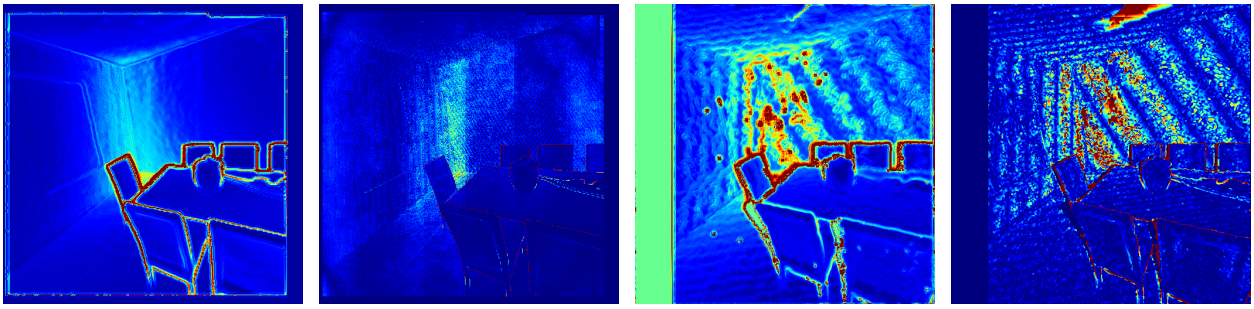


Figure 4.8: Comparison plots between the original and sensor-fused approaches. Lower opacity ellipses show the standard deviation of the metrics. **(Top)** accuracy v. completion, lower left is better. SGM result is cropped from view due poor results. **(Bottom)** L1 Rendering vs. completion ratio, lower right is better.

Visually, we see that the combination of PSMNet and SGM outperform their constituent results quite clearly. The results that fuse the SL model also tend to be clustered together, in part due to being dominated by a more trustworthy sensor. Despite this, we see relatively significant performance gains when we fuse the SL model and PSMNet depth maps within our uncertainty-aware extension of NICE-SLAM. Given more similarly balanced noise models, one might expect the potential performance gains to mirror what we see in the fusion of PSMNet and SGM, where there is a clear performance improvement across 3D reconstruction and 2D rendering.

In Figure 4.9, we compare the learned uncertainty to the true errors during the fusion of SGM and SL depth maps. We can see that the learned uncertainty for a fusion of SGM and SL depth maps appears

more reflective of the true errors, or proxy errors, which are calculated as the difference between the noisy and ground-truth depth maps. We note that these uncertainties are implicitly learned in an online fashion, without pre-training or pre-determined priors.



(a) Learned SL (b) True SL Error (c) Learned SGM (d) True SGM Error

Figure 4.9: Rendered uncertainty comparison of the SL-SGM fusion for scene Room 2.

Visually, we see correlation between areas of high uncertainty in the learned and true uncertainty. The corner of the room is an area of high uncertainty for the SL model, reflecting the true SL error. Band effects can be seen in the SGM approach that are captured in the learned SGM uncertainty.

Chapter 5

Discussion

5.1 Analysis of Results

In this work, we explore the construction of an uncertainty-aware neural implicit approach for 3D reconstruction and SLAM. We have presented detailed ablations using a single-sensor approach and cursory results from a multi-sensor implementation. Qualitatively, we also find informative uncertainty maps that reflect the true error in our noisy depth map models. This work achieves generally positive gains across metrics and was able to implicitly learn uncertainty of a given depth map in an online fashion. In contrast to other methods—*e.g.* SenFuNet [35] or RoutedFusion [44]—which pre-learn uncertainty through supervision with ground-truth depth maps, our approach is able to generate uncertainty maps without access to ground-truth.

5.1.1 Single-Sensor Approach

Within the single sensor-sensing environment, we showcase that several approaches toward learning uncertainty are viable. In particular, we use both an approach that only takes in information from the current image frame and an approach that leverages the structure of NICE-SLAM's 3D grid of features. Both of these methods, given the right feature inputs and parameters, are able to achieve statistical significant improvements on our evaluation metrics without any supervision over those metrics. Overall, the improvements achieved over the original implementation are small, but significant. This is not surprising, however, as it follows the work of Kendall and Gal [19], where they found small consistent improvements to computer vision tasks when incorporating uncertainty into their neural network architecture.

We find that in 3D reconstruction without tracking, the 3D grid of features using a larger regularizer appears to achieve more statistically significant improvements (see Appendix C). Such an approach accumulates observations from past frames in a continuous representation. One such interpretation of accumulating uncertainty over image frames can be seen loosely as informing a *prior* on the uncertainty rendering. Such an approach is necessarily camera pose dependent, however, and presents additional challenges with uncertainty coupling when tracking in enabled.

Learning uncertainty from a single source shows limited improvements and overall utility. But showcasing the possibility of learning uncertainty from a single source opens up many future directions for sensor fusion and learned weighting of different sensors exhibiting different noise distributions. Neural implicit scene representation is a still immature field and has yet to explore the full spectrum of advances seen by more standard representations like meshes and voxels. By capturing uncertainty with our method, we extend the flexibility of the NICE-SLAM framework and provide new information to leverage and further

close these gaps with classical methods by scaling depth observations by their uncertainty when generating the scene representation.

5.1.2 Two-Sensor Extension

We showcase that we can learn reasonable representations of uncertainty in an online fashion and without supervision of ground-truth depths to weight the reliability of two sensors. In particular, we highlight the consistent performance gain in *rendering*. Volume rendering is the backbone of NICE-SLAM and has beneficial evaluation properties when compared with the 3D metrics. In particular, the 2D rendering provides a direct comparison relying solely on the camera pose and makes no assumptions about the bounds of the scene. With the 3D metrics, the use of an input-dependent convex hull during mesh generation presents problems if there are large spurious errors in the 2D depth maps. For example, the SGM method has large spurious depth errors that result in an enlarged convex hull that leads to its low performing metrics. Further analysis can leverage the ground-truth depths to provide a consistent bounding hulls for 3D metric comparisons.

We find that our uncertainty-aware extension is capable of fusing two aligned depth maps and approach or exceed the performance of using individual sensors. Given further extensions to this approach—*e.g.* outlier rejection, colour weighting, asynchronous sensors—, we can expect to achieve better performance and/or greater flexibility.

5.2 Assumptions

We will dedicate some effort to address potential concerns regarding the construction of our loss function. In particular, we aim to address concerns regarding the independence of each sensor observation and of the specific conditions for separating out uncertainty into the aleatoric and epistemic categories.

5.2.1 Loss Function Assumptions

We make several assumptions when constructing our loss function in Section 3.2. These assumptions involve the error distribution and the pixel-wise independence. For the first point, we have assumed a symmetric Laplacian distribution in our assumption—*i.e.* that the probability of underestimating the depth and overestimating the depth is equal. Given that vision is occlusion-aware and scene detail is depth-dependent for vision sensors, we make a strong assumption on the error symmetry of the error. We additionally assume a particular steepness in the distribution fall-off when we assume a Laplacian over a Gaussian or other distribution. For the second point, we assume that depth measurements are I.I.D. for each given pixel. Given that time-of-flight sensors experience multi-path interference and structured-light or stereo-based methods rely on neighbouring points for detail extraction, we make a very strong assumption—and one that is certainly not fully accurate—in the independence of each pixel.

Nevertheless, we believe these are *reasonable* assumptions for a number of factors. In many modelling works, Gaussian or Laplacian noise is commonly employed to great effect, with the ability to generalize and perform well even when they differ from real-world distributions [6]. We also show later in Section 5.3.2 minor imbalances in the error distributions, but note that the distributions are close to symmetric. For the independence of pixel sampling, we make two arguments. First, we argue that the depth measurements are most likely affected by the local proximity of, or neighbouring, pixels. This means independence is likely for one pixel compared to the majority of other pixels due to the expectation that errors are only locally situated. Second, NICE-SLAM utilizes a sparse sampling methodology that results in a sparse selection of rays in the scene. This means that neighbouring rays are rarely selected, also improving the likelihood of independence if errors are, as we suspect, only locally correlated.

As such, we believe the underlying I.I.D. and distribution assumptions used in constructing our loss function are justified and not so distant from reasonable real-world conditions.

5.2.2 Uncertainty Assumptions

In Section 3.2, we present two major sources of uncertainty: aleatoric and epistemic. In this work we have focused on the aleatoric uncertainty, or uncertainty in the observation. However, there are many ways to classify uncertainty and uncertainties can be tightly coupled. We make an assumption that we can generate an unbiased depth rendering that allows us to retrieve the aleatoric uncertainty implicitly from our loss function. In practice, the depth rendering is model-dependent and, more specifically, dependent on the point sampling strategy. The default sampling strategy in NICE-SLAM samples points more densely around the depth measurement. For the majority of pixels, we expect that the depth measurement is close to the true surface. However at discontinuities and at spurious points, we may see large differences between the depth measurement and the true surface. In such cases, the depth rendering is directly affected by the measurement error. We assume that the decision to sample points more densely around the measurement will appropriately capture the distribution of possible depth readings. Edge locations might have, by nature, bimodal distributions for possible depth readings. Edges are relatively rare in depth maps, however, and we would expect that the majority of observations would not be affected by the pixel sampling strategy.

While we motivate our work in learning the aleatoric uncertainty implicitly, we must acknowledge that uncertainty is a complicated task to unravel. Drawing discrete boundaries between different sources of uncertainty can provide useful theoretical framing, but may not hold fully in practice.

5.3 Challenges in Learning Uncertainty

5.3.1 Sparse Pixel Sampling

NS-Replica has 1200×680 resolution which contains 816,000 pixels. With 5 keyframes, we have 4,080,000 pixels to sample from. Within NS-Replica, we sample 5000 pixels for mapping, or 0.12% of all available pixels. SFN-Replica has 512×512 resolution which contains 262,144 pixels. With 5 keyframes, we have 1,310,720 pixels to sample from. Within SFN-Replica, we sample 5000 pixels, or 0.38% of all available pixels. As a result of this pixel sampling strategy, we utilize only a very small selection of the total incoming data at each iteration of a mapping step and will rarely sample adjacent pixels for acquiring local context in the depth estimation.

Even ensuring that 5000 unique pixels are selected at every iteration in the 60 iteration mapping process, only a partial sampling of the entire frames is achieved at each mapping step. With this limited subset of pixels used in defining the scene, the model may overfit to the sparse set of observation provided. Learning the noise distribution might necessitate greater use of the incoming sensor information to prevent overfitting of the scene. On the other hand, the use of large voxel blocks at (16 cm) provides some regularizing effect on the scene representation and may mitigate some concern over the overfitting to the sparse points.

Additionally for each depth estimation, we only have a single observation. That is, for each pixel in the depth map we render, only a single pixel observation exists. To evaluate uncertainty using classical statistics, multiple observations is typically required to quantify uncertainty, especially if we are working under the assumption that each pixel is I.I.D. As such, we would expect the task of determining uncertainty to be fairly challenging within the NICE-SLAM framework.

5.3.2 Poorly-balanced Error Distributions from Pixel Selection

One concern when trying to train for a network that learns uncertainty is the uneven distribution of low and high error pixels. Even within a noisy depth map, most pixels are relatively accurate, providing an imbalanced dataset. This imbalance can lead to the network overfitting to the low uncertainty regions and failing to learn the high uncertainty representations. We investigate the error distribution within the SFN-Replica dataset and find that the errors are highly concentrated around the zero-error region, shown in Figure 5.1.

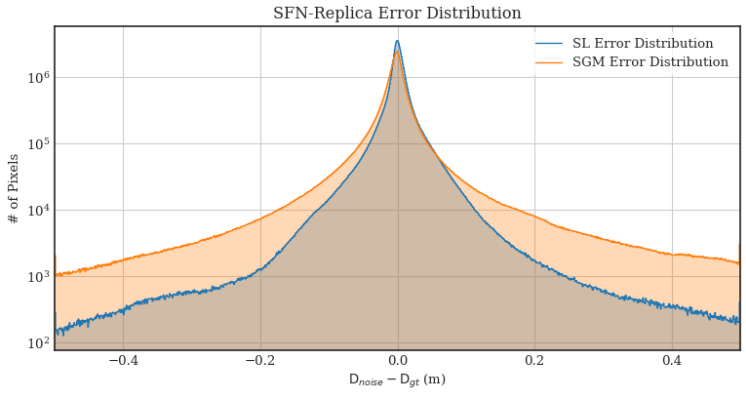


Figure 5.1: Pixel-wise error distribution of noisy depth maps presented in log-scale.

We note that the distribution falls from 10^6 to 10^3 number of samples when we look at pixels with an

error of $0\ \mathrm{m}$ and with an error of $0.4\ \mathrm{m}$. When we look at the CDF of the error distributions, we find that $> 90\%$ of pixels in the SL noisy depth maps and $> 80\%$ of pixels in the SGM stereo depth maps exhibit less than $5\ \mathrm{cm}$ of error. We present the CDF of these two distribution is Figure 5.2.

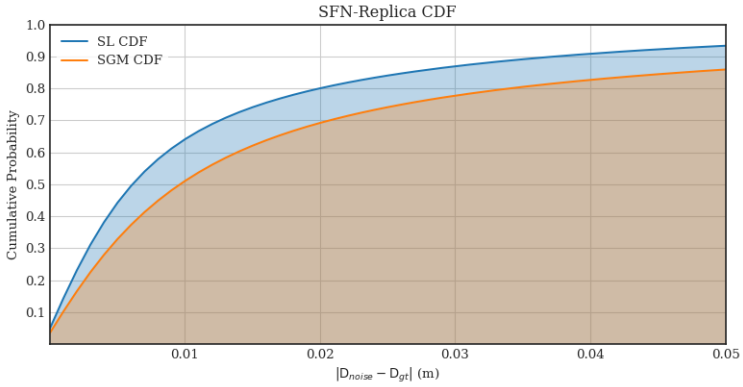


Figure 5.2: CDF of SFN-Replica error distribution.

This sampling issue of low and high error pixels—*i.e.* the dataset imbalance between high and low uncertainty "classes"—was also discussed in the work by Bae *et al.* [2] in investigating surface normal uncertainty. They formulate a refinement step that uses an MLP that refines a set of pixel-wise input features that had previously been used for a coarse prediction. To train these MLP refinement modules, the authors employ *uncertainty-guided sampling* which selects from the highest uncertainty pixels and appends these samples to uniformly sampled pixels. This approach prevents bias in training towards large planar surfaces and improves the surface normal reconstruction.

Such an approach to learn the uncertainty in a more balanced fashion would be beneficial. However, we aim to learn uncertainty in an online manner and without direct supervision from the GT. To incorporate balanced training would require changes in our underlying assumptions and significant reworking of the training pipeline and architecture. As an avenue of exploration, such an approach is promising and we hope that we can use this insight to inform future changes of our architecture.

5.4 Future Work

In the design of NICE-SLAM or any SLAM pipeline, there are many practical considerations that can impact overall performance. There are many avenues for the future direction of this work, including the extension of our framework to incorporate different sensor systems.

5.4.1 Extension to Colour Sensors

In this work, we isolate our investigation to looking at just depth maps in a single-sensor or two-sensor environment. A colour or RGB image provides additional information and can be unified within our existing framework. Making the same assumptions as described in Section 3.2, we can formulate an equivalent probabilistic function describing each pixel observation. Assuming each RGB pixel is I.I.D., we can describe the probability of the current observations as shown in Eq. (5.1).

$$\begin{aligned} P(C_1, ..., C_M) &= P(C_1)...P(C_M) \\ &= \prod_{m=1}^{M} P(C_m) \\ &= \prod_{m=1}^{M} \frac{1}{2\beta_{C,m}} e^{-\frac{|C_m - \mu_{C,m}|}{\beta_{C,m}}} \end{aligned} \tag{5.1}$$

where C_i is the RGB values of pixel i. Combining with the original depth maps, we can construct a joint probability distribution using both depth maps and RGB images. We present this formulation in Eq. (5.2).

$$\begin{aligned} P(D_1, ..., D_M, C_1, ..., C_M) &= P(D_1)...P(D_M) \times P(C_1)...P(C_M) \\ &= \prod_{m=1}^{M} P(D_m)P(C_m) \\ &= \prod_{m=1}^{M} \frac{1}{2\beta_{D,m}} e^{-\frac{|D_m - \mu_{D,m}|}{\beta_{D,m}}} \times \frac{1}{2\beta_{C,m}} e^{-\frac{|C_m - \mu_{C,m}|}{\beta_{C,m}}} \end{aligned} \tag{5.2}$$

Converting the above formulation into a loss function, we can utilize the colour rendering methods originally used in NICE-SLAM. We do, however, also append an additional network to learn the colour uncertainty in the same manner as the depth uncertainty described in this work. Consequently, we can construct the loss function as show in Eq. (5.3).

$$\mathcal{L} = \sum_{m=1}^{M} \left(\frac{|D_m - \hat{D}_m|}{\hat{\beta}_{D,m}} + \log(\hat{\beta}_{D,m}) + \frac{|C_m - \hat{C}_m|}{\hat{\beta}_{C,m}} + \log(\hat{\beta}_{C,m}) \right) \tag{5.3}$$

where $\hat{D}$, $\hat{C}$, and $\hat{\beta}$ are parameterized using the deep learning architectures described in Section 3.4. This approach has the capacity to learn the weighting function between sensor observations, independently of hand-tuning results. In the original NICE-SLAM, the implementation selects specific values to weight the depth and colour loss, which operate using different scales. Automatically learning an effective scaling between the two would be highly beneficial in online SLAM environments.

5.4.2 Extension to Multiple Non-aligned Sensors

The previous section describes how we could combine the RGB and depth sensors of a unified RGBD system. Beyond such a system, it is easy to extend the approach from aligned sensors to rigidly offset sensors, or to even use different data inputs. For example, a stereo pair of depth cameras could be employed and sampled independently, generating an equivalent formulation from the single sensor sampled case. Alternatively, if we are provided a sparse point cloud, we could sample points and regress depth in the same fashion as our dense depth sensors we have investigated so far, allowing for scene interpolation and depth completion.

Given two synchronized and rigidly offset sensors, we can sample a set of observations $a \in \{1, ..., A\}$ from depth sensor A and a set of observations $b \in \{1, ..., B\}$ from depth sensor B. The corresponding generalized loss function could be described as shown in Eq. (5.4).

$$\mathcal{L} = \sum_{a=1}^{A} \left(\frac{|D_a - \hat{D}_a|}{\hat{\beta}_{D,a}} + \log(\hat{\beta}_{D,a}) \right) + \sum_{b=1}^{B} \left(\frac{|D_b - \hat{D}_b|}{\hat{\beta}_{D,b}} + \log(\hat{\beta}_{D,b}) \right) \tag{5.4}$$

Extensions to non-synchronized and independently-moving sensors or agents are also possible, but should be more challenging due to the requirements for accurate tracking and balancing scene updates. In a synchronized setting, the different sensors could have a complimentary effect that improves sensor fusion during each update step. Asynchrocity requires some degree of input batching to allow for joint scene optimization. Ultimately, we believe that the current implementation of our uncertainty-aware module is flexible and allows us to explore the potential in multi-sensor and multi-agent situations in a motivated and self-balancing manner. These concepts we introduce here are not explored in this work, but are trivial conceptual extensions based on the flexibility of how we motivate our approach in implicitly learning uncertainty in an online fashion.

Chapter 6

Conclusion

In this work, we have shown that we can learn the uncertainty of a depth sensor in an online fashion without supervision from the ground-truth mesh or depth maps. We introduce a theoretical framework for how to model individual pixel observations in a probabilistic manner given strong assumptions about independence. From this framework, we construct a motivated objective function to learn uncertainty implicitly. This use of uncertainty within a single-sensor environment can improve 3D reconstruction and localization within the neural implicit framework of NICE-SLAM. We find evidence that our approach of using 2D feature maps and volume rendering through a persistent 3D grid of features is capable of improving tracking, reconstruction, and rendering metrics with statistical significance.

Furthermore, we find that this work naturally extends to multi-sensor systems. We showed that we can fuse the depth information from two aligned sensors to improve the rendering results and some 3D reconstruction metrics. We also find that the performance approaches or exceeds the performance of the more reliable sensor, showing promise in further work to refine the fusion of independent sensors in 3D reconstruction.

We believe this works furthers our understanding of how to leverage uncertainty to improve the results within neural implicit representations. This work naturally leads to possible extension involving the fusion of different sensing modalities, such as RGB images, and from other sensor types, such as lidar. Additionally, this work provides the groundwork for extensions into non-aligned and asynchronous fusion of multiple sensors which have applications in multi-agent and multi-sensor robotics.

Appendix A

Environment & Code Repository

The codebase used for this project can be found here: https://gitlab.ethz.ch/kevita/uncertainty-nice

A.1 System Description

This work was primarily conducted on the CVL SLURM cluster for students. The deep learning training of NICE-SLAM and our associated extension was performed using `titan_x`, `titan_xp`, or `geforce_gtx_titan_x` GPUs.

To run the evaluation scripts on the cluster, headless rendering in Open3D is required to be built from its source code. This includes headless support in OS Mesa. We provide instructions in the repository on how to set-up the evaluation scripts to work on the displayless clusters.

A.2 Code Repository

This work builds off the NICE-SLAM repository (github.com/cvg/nice-slam) with modification and improvements. The NICE-SLAM code-base is a challenging code-base to work with due to its non-deterministic outputs that require statistical methods for proper evaluation. Additionally, there are potential improvements to the code's modularity and extensibility that we hope to address.

The repository provides an installation guide and environment configuration file in our working repository for our NICE-SLAM extension, managed using Conda. A summary of key libraries and their active version can be found below:

- `Python = 3.7.11`
- `pytorch = 1.11.0`
- `opencv = 4.5.5.64`
- `numpy = 1.21.5`

Within the repository, we provide the modified NICE-SLAM code in the `nice-slam/` folder. We also provide various scripts or data visualization and data preparation in the `src/prepare_dataset/` folder, such as the code for inducing noise into depth maps. The `src/habitat/` folder contains the files necessary to render a sequence of stereo depth maps and stereo images based on trajectories provided in either the

NICE-SLAM Replica (NS-Replica) or SenFuNet Replica (SFN-Replica) formats. The `notebooks/` folder contains some working Jupyter notebooks used for prototyping, data analysis, and data visualization. The `docs/` folder contains instructions on how to run and set-up the various auxiliary scripts provided.

A.3 Stochasticity & Non-determinism

The default implementation of NICE-SLAM is non-deterministic. To aid in development, we investigated different parameters in an effort to achieve repeatable results from the base NICE-SLAM implementation.

According to Pham *et al*. [30], there is significant variance when re-training deep learning models. These are categorized into algorithmic sources or implementation sources. A list of these factors can be found below:

- *Algorithmic* –non-deterministic DL layers, weight initialization, data augmentation, batch ordering
- *Implementation* –parallel processes, auto-selection of primitive operations, scheduling floating-point precision

In an effort to achieve deterministic outputs, we initialize the random number generator (RNG) seed in the `pytorch`, `numpy`, and `random` libraries. This leads to the initial forward passes to generate the same results, except for minor differences due to parallel floating point operations. This alone was insufficient to achieve deterministic outputs.

We next attempt to force `pytorch` to select deterministic algorithms for computation by setting the `use_deterministic_algorithms()` internal flag to `True`. Unfortunately, the backward pass of the `grid_sample()` function is non-deterministic and has no deterministic implementation in pytorch. This function is used for interpolating from the voxel grid of features in the decoder. We note that a deterministic implementation of backwards pass through trilinear interpolation should be possible. As such, our efforts shifted from ensuring deterministic outputs, to controlling our analysis process to accommodate the stochasticity of the output.

A first strategy to address the variance in output is simple aggregate statistics across a number of runs. We are particularly interested in the mean result and the accompanied standard deviation of the repeated runs. In our experiments, we find larger variances in results using the original implementation of NICE-SLAM, especially in the case where tracking is active, as we evaluate on noise-induced depth maps. Using the original tracking loss, we can find standard deviations greater than $\pm 50\%$ of the metric mean result due to tracking failures.

A second strategy to determine the significance of our work is through a unpaired t-tests that tells us the likelihood that both results exhibit the same mean. Such tests give us some insight into the effectiveness of various implemented changes within the NICE-SLAM framework. We present the evaluation method in more detail in Chapter 4.

Appendix B

Weighting Bias in Volume Rendering

As pointed out in NeuS [43], the weighting function employed in volume rendering is inherently *biased*. Despite this, an occupancy model that exhibits sharp transitions from free space to occupied space can mitigate the effects of bias.

B.1 Depth Rendering

We show that, given an accurate uncertainty model, we can reconstruct the estimated uncertainty at the specific estimated depth. We show this result by comparing a few example cases where we simulate the alpha-composition process using:

1. An occupancy oracle based on a logistic curve.
2. A depth-dependent uncertainty function based on a polynomial (squared) function.

First, we showcase an example where we utilize equally-spaced point sampling on a gradual, or more uncertain, occupancy boundary. Figure B.1 shows that, due to the shallowness of the surface boundary, the extraction of the surface is biased. The red dotted line $D_e(p)$ shows the rendered depth, the blue line $O(x)$ is the occupancy oracle, and the green markers $W(x)$ are the associated weighting values. N_s represents the number of points sampled.

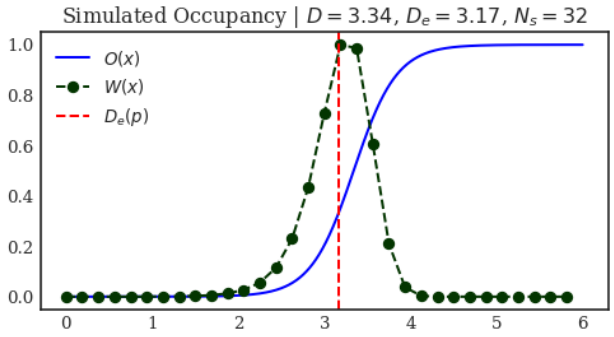


Figure B.1: Depth determination from equally-spaced point sampling in alpha-composition over a less certain surface boundary.

NICE-SLAM optimizes for the scene geometry and naturally minimizes this model boundary uncertainty over time. Figure B.2 shows how a higher certainty model boundary leads to more accurate retrieval of the surface boundary.

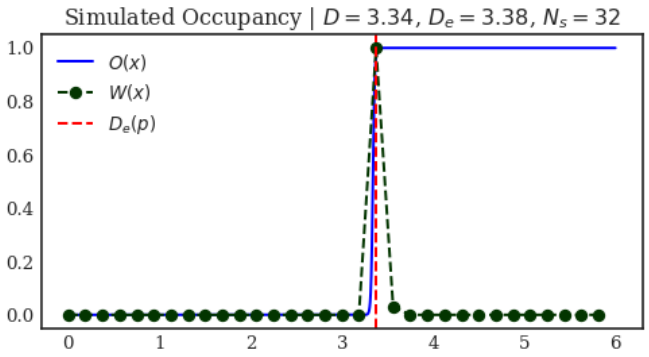


Figure B.2: Depth determination from equally-spaced point sampling in alpha-composition over a more certain surface boundary.

Another addition that NICE-SLAM uses is informed sampling where N_i points are sampled near the depth measurement. In the previous example, we see that the sharp scene bounds leads to better surface identification. The inclusion of informed sampling—that is equally-spaced sampling in the range of $[0.95D_m, 1.05D_m]$ where D_m is the measured depth—allows for the surface bound to be better identified. Figure B.3 shows how informed sampling can further reine the depth retrieval.

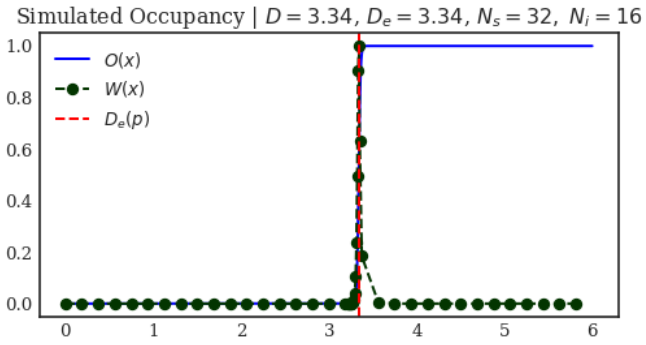


Figure B.3: Depth determination from equally-spaced point sampling in alpha-composition over a more certain surface boundary with informed sampling.

B.2 Uncertainty Rendering

We can also show that an alpha-composited uncertainty is able to retrieve results close to the oracle uncertainty at the estimated depth. Fig. B.4 shows this result where we can compare the true depth D and true uncertainty $S(D)$ to the estimated depth D_e and volume rendered uncertainty S_e.

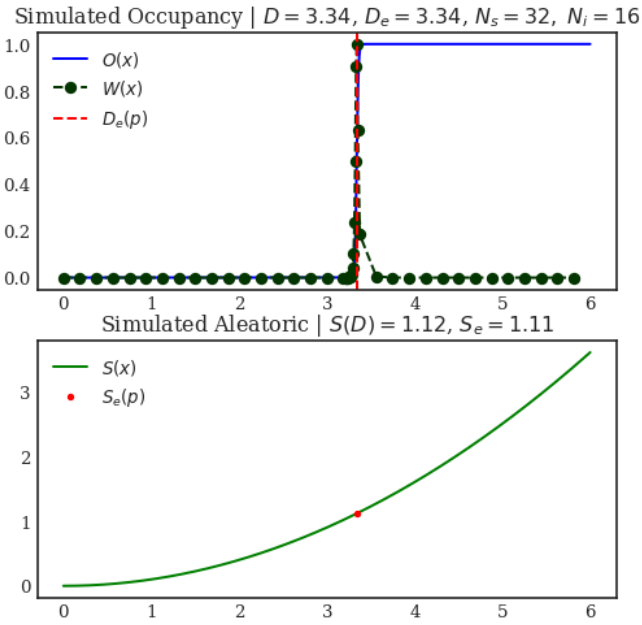


Figure B.4: Uncertainty and depth estimation from alpha composition.

Appendix C

Architecture Ablation Significance

We perform a cursory analysis on the total number of improved metrics in Sections 4.5.1 and 4.5.2, showing the general trends of improvement. We now present the detailed results of the significance analysis of the various ablations. The summary may be found in Table C.1.

Table C.1: Statistical significance of ablation differences based on Welch's t-test. *Italicized* results are not statistically significant ($P > 0.05$). Grayed results degraded using our uncertainty-aware loss.

		2D Features				**3D Grid**			
Trial	Scene	P Acc.	P Comp.	P Ratio	P 2D	P Acc.	P Comp.	P Ratio	P 2D
1K7FS	Office 0	*65.8%* ↑	2.7% ↑	*6.8%* ↑	*36.9%* ↑	*57.0%* ↓	1.4% ↓	*82.3%* ↑	0.0% ↓
	Office 1	*91.2%* ↓	*62.4%* ↑	*46.7%* ↑	*41.1%* ↑	*24.2%* ↓	*12.1%* ↑	*11.1%* ↑	*24.0%* ↑
	Room 2	*33.6%* ↓	4.1% ↓	*31.2%* ↓	*22.4%* ↓	*73.0%* ↓	0.0% ↓	1.9% ↓	0.7% ↓
1K2FS	Office 0	*20.1%* ↑	1.2% ↓	*12.9%* ↓	0.0% ↓	1.8% ↑	*43.4%* ↓	*66.7%* ↑	0.0% ↓
	Office 1	*21.0%* ↑	*13.3%* ↑	*13.9%* ↑	*33.1%* ↓	*97.0%* ↓	*39.8%* ↑	*64.5%* ↑	*20.3%* ↓
	Room 2	*11.3%* ↓	0.0% ↓	0.0% ↓	0.0% ↓	*10.1%* ↓	0.0% ↓	0.1% ↓	1.1% ↓
1K7FL	Office 0	*6.7%* ↓	*71.1%* ↑	*15.9%* ↑	*7.1%* ↓	*16.2%* ↑	*46.3%* ↓	*45.8%* ↓	*81.5%* ↓
	Office 1	*51.3%* ↓	*61.8%* ↑	*89.0%* ↑	*42.3%* ↑	*9.8%* ↓	2.5% ↑	2.2% ↑	*62.3%* ↑
	Room 2	*62.3%* ↓	*94.4%* ↓	*81.7%* ↓	*61.9%* ↓	3.8% ↑	*60.0%* ↑	*52.8%* ↑	1.2% ↑
1K2FL	Office 0	0.5% ↑	0.% ↑	0.2% ↑	*78.0%* ↑	*9.2%* ↑	0.% ↑	0.1% ↑	2.3% ↑
	Office 1	*39.5%* ↓	*5.1%* ↑	3.4% ↑	*35.7%* ↓	*43.7%* ↑	0.4% ↑	0.2% ↑	*41.1%* ↓
	Room 2	*16.3%* ↑	*47.9%* ↑	*37.2%* ↑	*20.2%* ↑	*23.6%* ↑	*24.6%* ↑	*13.5%* ↑	1.7% ↑
5K7FS	Office 0	*24.9%* ↓	*61.4%* ↓	*78.4%* ↑	0.0% ↓	*40.6%* ↓	0.3% ↓	*10.0%* ↓	0.0% ↓
	Office 1	*87.5%* ↓	*19.2%* ↑	*12.7%* ↑	*34.0%* ↓	*6.3%* ↑	*6.8%* ↑	*41.2%* ↑	*5.2%* ↑
	Room 2	*18.3%* ↓	0.0% ↓	0.0% ↓	*28.7%* ↓	3.9% ↑	0.0% ↓	0.0% ↓	0.4% ↓
5K2FS	Office 0	0.7% ↑	*26.7%* ↑	*10.2%* ↑	*17.1%* ↑	*11.7%* ↑	0.9% ↓	*38.9%* ↑	1.9% ↓
	Office 1	*89.1%* ↓	3.6% ↑	3.7% ↑	*57.9%* ↑	*67.9%* ↑	*14.1%* ↑	*18.1%* ↑	*8.3%* ↑
	Room 2	*50.9%* ↑	*63.8%* ↓	*55.6%* ↑	*88.8%* ↑	*93.2%* ↑	0.0% ↓	0.1% ↓	0.1% ↓
5K7FL	Office 0	*29.6%* ↓	*10.8%* ↑	*14.8%* ↑	*18.4%* ↓	*9.1%* ↑	*18.1%* ↑	*11.9%* ↑	0.8% ↑
	Office 1	*48.7%* ↑	2.4% ↑	0.3% ↑	*80.1%* ↑	*72.1%* ↓	2.3% ↑	4.4% ↑	*41.2%* ↑
	Room 2	60.4% ↓	*30.6%* ↑	*14.0%* ↑	*67.2%* ↑	3.9% ↑	*16.8%* ↑	*57.7%* ↓	*36.9%* ↑
5K2FL	Office 0	*75.0%* ↓	*16.8%* ↑	*8.6%* ↑	*61.8%* ↑	*7.3%* ↑	*87.6%* ↑	*15.2%* ↑	*48.4%* ↑
	Office 1	*25.0%* ↓	0.9% ↑	0.1% ↑	*14.3%* ↓	*52.4%* ↓	3.6% ↑	0.6% ↑	3.9% ↑
	Room 2	*78.9%* ↓	*22.5%* ↓	*67.0%* ↓	*44.4%* ↑	0.1% ↑	*32.2%* ↑	*24.7%* ↑	*23.5%* ↑

We can see that many of the results either show degradation or are not statistically significant. We discussed briefly in the main text the quantity of improved and degraded metrics. However, this does not take into account whether a difference in metric is statistically significant. We present in Table C.2 the number of significant improvements, the number of significant degradations, and the *net total* of significant improvements.

Table C.2: Summary of the effect of different ablations in understanding the effect of different uncertainty-aware architectures.

Run	Architecture	Patch-size	Features	β_{min} [m]	Sig ↑	Sig ↓	Net Sig
2D1K7FS	2D Ray	1	7	1e-3	1	1	0
2D1K2FS	2D Ray	1	2	1e-3	0	5	-5
2D1K7FL	2D Ray	1	7	1e-1	0	0	0
2D1K2FL	2D Ray	1	2	1e-1	3	0	3
2D5K7FS	2D Patch	5	7	1e-3	0	3	-3
2D5K2FS	2D Patch	5	2	1e-3	3	0	3
2D5K7FL	2D Patch	5	7	1e-1	2	0	2
2D5K2FL	2D Patch	5	2	1e-1	2	0	2
3D1K7FS	3D Grid	1	7	1e-3	0	5	-5
3D1K2FS	3D Grid	1	2	1e-3	1	4	-3
3D1K7FL	3D Grid	1	7	1e-1	4	0	4
3D1K2FL	3D Grid	1	2	1e-1	6	0	6
3D5K7FS	3D Grid	5	7	1e-3	1	5	-4
3D5K2FS	3D Grid	5	2	1e-3	0	5	-5
3D5K7FL	3D Grid	5	7	1e-1	4	0	4
3D5K2FL	3D Grid	5	2	1e-1	4	0	4

We see that the selected 3D method, "3D1K2L," achieves the best performance with six significant improved results and no significant degradations. We also note that all of the methods employing the 3D feature grid utilizing a small regularizer have net negative improvements. The two methods we decided between for the 2D image feature MLPs each have three significant improvements and no significant degradations.

Appendix D

Multi-Sensor Extension Code Modifications

To accommodate the two-sensor input, some major changes were required within the NICE-SLAM code repository. We detail the major changes here.

D.1 Dataset Loader

The dataset loader is used to ingest images, depth maps, and their ground-truth poses from a sequential capture of a scene.

Return Types. To accommodate the inclusion of various feature maps and the acquisition of a second depth map, we alter the return structure from each individual element, to a dictionary of elements. This provides extensibility for adding additional features or additional synchronized depth maps.

D.2 Mapper

The mapper runs as one of the main processes in NICE-SLAM and is responsible for updating the scene representation with the provided depth map and image information.

Frustum Feature Selection. When provided with more than one input sensor, the features set for optimization extends to the furthest depth reading across sensors for a single pixel. This ensures that the optimizable parameters include the set of feature points in the grid that would theoretically be observed by any sensor.

Keyframe Selection. We modify the keyframe selection script to accept an *averaged* depth map between the two input sensors for determining the relevant keyframes in the "bundle adjustment" mapping process. A future extension could include using the uncertainty maps to form a weighted depth map for this purpose.

Training Stages. The original NICE-SLAM optimizes each mapping step by progressively adding more detail. In our implementation of uncertainty-aware single-sensor optimization, we utilize the default loss for training the middle stage. In our two-sensor implementation, we weight each sensor observation by the frozen uncertainty head throughout the middle stage. This difference arises from the fact that the single-sensor environment has fewer observations and we need to initialize the middle grid more consistently with coarse geometry.

D.3 Renderer

The renderer class is responsible for rendering selected pixels through the volume rendering pipeline, including the forward rendering through the nerual network decoders for occupancy, colour, and uncertainty.

Point Sampling. The original implementation of NICE-SLAM samples 16 points near the depth measurements as well as 32 points a long the ray. With the addition of another sensor, we sample 16 points at around each sensor measurement on top of the 32 points along the ray for a total of 64 points. The 32 points sampled equally throughout the ray are determined by the maximum depth of both sensors.

Bibliography

[1] Moloud Abdar, Farhad Pourpanah, Sadiq Hussain, Dana Rezazadegan, Li Liu, Mohammad Ghavamzadeh, Paul Fieguth, Xiaochun Cao, Abbas Khosravi, U. Rajendra Acharya, Vladimir Makarenkov, and Saeid Nahavandi. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion*, 76:243–297, 2021. 7

[2] Gwangbin Bae, Ignas Budvytis, and Roberto Cipolla. Estimating and Exploiting the Aleatoric Uncertainty in Surface Normal Estimation. In *the proceedings of the International Conference on Computer Vision (ICCV)*, 2021. 8, 48

[3] Gwangbin Bae, Ignas Budvytis, and Roberto Cipolla. IronDepth: Iterative Refinement of Single-View Depth using Surface Normal and its Uncertainty. In *the proceedings of the British Machine Vision Conference (BMVC)*, 2022. 8

[4] Jonathan T. Barron and Jitendra Malik. Intrinsic scene properties from a single RGB-D image. *Transactions on Pattern Analysis and Machine Intelligence*, 38(4):690–703, 2016. 23

[5] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. SURF: Speeded Up Robust Features. In *the proceedings of the European Conference on Computer Vision (ECCV)*. CVF, 2006. 5

[6] David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational Inference: A Review for Statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017. 45

[7] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. BRIEF: Binary robust independent elementary features. In *the proceedings of the European Conference on Computer Vision (ECCV)*. CVF, 2010. 5

[8] Carlos Campos, Richard Elvira, Juan J.Gomez Rodriguez, Jose M.M. Montiel, and Juan D. Tardos. ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial, and Multimap SLAM. *IEEE Transactions on Robotics*, 37(6):1874–1890, 2021. 5

[9] Yan Pei Cao, Leif Kobbelt, and Shi Min Hu. Real-time high-accuracy three-dimensional reconstruction with consumer RGB-D cameras. *ACM Transactions on Graphics*, 37(5), 2018. 6

[10] Jia Ren Chang and Yong Sheng Chen. Pyramid Stereo Matching Network. In *the proceedings of Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE/CVF, 2018. 23

[11] Brian Curless and Marc Levoy. Volumetric method for building complex models from range images. In *the proceedings of the SIGGRAPH Conference on Computer Graphics*. ACM, 1996. 5, 7

[12] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. ScanNet: Richly-annotated 3D reconstructions of indoor scenes. In *the proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE/CVF, 2017. 24

[13] Frank Dellaert. NeRF Explosion 2020, 2020. 6

[14] Frank Dellaert. NeRF at ICCV 2021, 2021. 6

[15] Frank Dellaert and Andrew Marmon. NeRF at CVPR 2022, 2022. 6

[16] Ankur Handa, Thomas Whelan, John McDonald, and Andrew J. Davison. A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM. In *the proceedings of the International Conference on Robotics and Automation*. IEEE, 2014. 23

[17] Heiko Hirschmüller. Stereo processing by semiglobal matching and mutual information. *Transactions on Pattern Analysis and Machine Intelligence*, 30(2):328–341, 2008. 23

[18] Krishna Murthy Jatavallabhula, Ganesh Iyer, and Liam Paull. GradSLAM: Dense SLAM meets Automatic Differentiation. In *the proceedings of the International Conference on Robotics and Automation (ICRA)*. IEEE, 2020. 6

[19] Alex Kendall and Yarin Gal. What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision? In *the proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*. NeurIPS Foundation, 2017. 8, 13, 14, 43

[20] Chen Hsuan Lin, Wei Chiu Ma, Antonio Torralba, and Simon Lucey. BARF: Bundle-Adjusting Neural Radiance Fields. In *the proceedings of the International Conference on Computer Vision (ICCV)*. IEEE/CVF, 2021. 7

[21] David G. Lowe. Object recognition from local scale-invariant features. In *the proceedings of the International Conference on Computer Vision (ICCV)*. IEEE/CVF, 1999. 5

[22] Ricardo Martin-Brualla, Noha Radwan, Mehdi S.M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth. NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections. In *the proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE/CVF, 2021. 6, 15, 16

[23] N. Max. Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1995. 6

[24] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In *the proceedings of the European Conference on Computer Vision (ECCV)*. CVF, 2020. 6, 9

[25] Raul Mur-Artal, J. M.M. Montiel, and Juan D. Tardos. ORB-SLAM: A Versatile and Accurate Monocular SLAM System. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015. 5

[26] Raul Mur-Artal and Juan D. Tardos. ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. *IEEE Transactions on Robotics*, 33(5):1255–1262, 2017. 5

[27] Richard A. Newcombe, Andrew Fitzgibbon, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J. Davison, Pushmeet Kohi, Jamie Shotton, and Steve Hodges. KinectFusion: Real-time dense surface mapping and tracking. In *the proceedings of the International Symposium on Mixed and Augmented Reality*. IEEE, 2011. 5, 7

[28] Michael Oechsle, Songyou Peng, and Andreas Geiger. UNISURF: Unifying Neural Implicit Surfaces and Radiance Fields for Multi-View Reconstruction. In *the proceedings of the International Conference on Computer Vision (ICCV)*. IEEE/CVF, 2021. 6

[29] Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional Occupancy Networks. In *the proceedings of the European Conference Computer Vision (ECCV)*. CVF, 2020. 6, 9

[30] Hung Viet Pham, Shangshu Qian, Jiannan Wang, Thibaud Lutellier, Jonathan Rosenthal, Lin Tan, Yaoliang Yu, and Nachiappan Nagappan. Problems and opportunities in training deep learning software systems: an analysis of variance. In *the proceedings of the International Conference on Automated Software Engineering (ASE)*, New York, NY, USA, 2020. IEEE/ACM. 54

[31] Konstantinos Rematas, Andrew Liu, Pratul P. Srinivasan, Jonathan T. Barron, Andrea Tagliasacchi, Thomas Funkhouser, and Vittorio Ferrari. Urban Radiance Fields. In *the proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE/CVF, 2021. 6

[32] Antoni Rosinol, John J. Leonard, and Luca Carlone. NeRF-SLAM: Real-Time Dense Monocular SLAM with Neural Radiance Fields. *arXiv*, 2022. 6, 7

[33] Edward Rosten and Tom Drummond. Machine Learning for High-Speed Corner Detection. In *the proceedings of the European Conference on Computer Vision (ECCV)*. CVF, 2006. 5

[34] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. ORB: An efficient alternative to SIFT or SURF. In *the proceedings of the International Conference on Computer Vision (ICCV)*. IEEE/CVF, 2011. 5

[35] Erik Sandström, Martin R. Oswald, Suryansh Kumar, Silvan Weder, Fisher Yu, Cristian Sminchisescu, and Luc Van Gool. Learning Online Multi-Sensor Depth Fusion. In *the proceedings of the European Conference Computer Vision (ECCV)*. CVF, 2022. 7, 23, 43

[36] Mohamed Sayed, John Gibson, Jamie Watson, Victor Prisacariu, Michael Firman, and Clément Godard. SimpleRecon: 3D Reconstruction Without 3D Convolutions. In *the proceedings of the European Conference Computer Vision (ECCV)*. CVF, 2022. 17

[37] Jianbo Shi and Carlo Tomasi. Good Features to Track. In *the proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 1994. 5

[38] R. Smith, M. Self, and P. Cheeseman. Estimating Uncertain Spatial Relationships in Robotics. In Ingemar J. Cox and Gordon T. Wilfong, editors, *Autonomous Robot Vehicles*, volume 4, pages 167–193. Springer, New York, NY, 1990. 5

[39] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J. Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, Anton Clarkson, Mingfei Yan, Brian Budge, Yajie Yan, Xiaqing Pan, June Yon, Yuyang Zou, Kimberly Leon, Nigel Carter, Jesus Briales, Tyler Gillingham,

Elias Mueggler, Luis Pesqueira, Manolis Savva, Dhruv Batra, Hauke M. Strasdat, Renzo De Nardi, Michael Goesele, Steven Lovegrove, and Richard Newcombe. The Replica Dataset: A Digital Replica of Indoor Spaces. *arXiv*, 2019. 23

[40] Jürgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. A benchmark for the evaluation of RGB-D SLAM systems. In *the proceedings of the International Conference on Intelligent Robots and Systems (IROS)*. IEEE/RSJ, 2012. 21, 23

[41] Edgar Sucar, Shikun Liu, Joseph Ortiz, and Andrew J. Davison. iMAP: Implicit Mapping and Positioning in Real-Time. In *the proceedings of the International Conference on Computer Vision (ICCV)*. IEEE/CVF, 2021. 7, 21, 23, 36

[42] Zachary Teed and Jia Deng. DROID-SLAM: Deep Visual SLAM for Monocular, Stereo, and RGB-D Cameras. In *the proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*. NeurIPS Foundation, 2021. 7

[43] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. NeuS: Learning Neural Implicit Surfaces by Volume Rendering for Multi-view Reconstruction. In *the proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*. NeurIPS Foundation, 2021. 6, 55

[44] Silvan Weder, Johannes L Sch, and Martin R Oswald. RoutedFusion : Learning Real-time Depth Map Fusion. In *the proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE/CVF, 2020. 7, 15, 34, 43

[45] Welch and Bernard Lewis. The generalization of 'Student's' problem when several different population variances are involved. *Biometrika*, 34(1-2):28–35, 1947. 21

[46] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R. Oswald, and Marc Pollefeys. NICE-SLAM: Neural Implicit Scalable Encoding for SLAM. In *the proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE/CVF, 2022. 2, 7, 21, 23, 24