Underwater 3D Visual Perception and Generation

Tianyi Zhang

CMU-RI-TR-25-82

August 18th, 2025

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee:
Matthew Johnson-Roberson, Carnegie Mellon University, *chair*
Oliver Kroemer, Carnegie Mellon University
Shubham Tulsiani, Carnegie Mellon University
Matthew O'Toole, Carnegie Mellon University
Katherine Skinner, University of Michigan, Ann Arbor

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Robotics.

Abstract

With modern robotic technologies, seafloor imagery has become more accessible to researchers and the public. This dissertation leverages deep learning and 3D vision techniques to deliver valuable information from seafloor image observations collected by robotic platforms.

Despite the widespread use of deep learning and 3D vision algorithms across various fields, underwater imaging presents unique challenges, such as lack of annotations, color distortion, and inconsistent illumination, which limit the effectiveness of off-the-shelf algorithms. This dissertation tackles the fundamental problem of building 3D representations from raw underwater images with heavy effects from light sources and medium interference. The following algorithms are developed to achieve seafloor 3D reconstruction with photorealistic quality: (i) Unsupervised underwater caustic removal with recurrent 3D Gaussian Splatting (ii) Deep water true color restoration with neural reflectance fields (iii) Camera-light source calibration for robotic platforms (iv) Dark environment relighting with 3D Gaussian Splatting. With the large amount of seafloor data as training data collected by robots, this dissertation further investigates the use of deep generative models to generate large-scale underwater terrains with natural spatial variance in appearance. The synthesized terrain can be integrated with the learned underwater lighting effects, to present realistic novel-view rendering results.

This dissertation shows examples of how 3D computer vision and deep generative models can be combined with physical laws, statistical principles, and foundation models, to address the unique challenges in underwater robotic perception. Collectively, these contributions lay the groundwork for reconstructing hi-fidelity underwater scenes to help people better understand benthic ecosystem and generating simulation environments that help close the sim-to-real gap in underwater robot perception.

Funding

This work is supported by NOAA under grant NA22OAR0110624.

Contents

When this dissertation is viewed as a PDF, the page header is a link to this Table of Contents.

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Lessons from the field

Humans have been exploring the benthic world towards deeper oceans over history. Early efforts on seafloor observation beyond the physical limit of the human body can be traced back to 4th century BC, as described by Aristotle that an equipment called diving bell was used to enable the diver to respire and observe the seafloor in an underwater cauldron with open bottom [8]. Such equipment was more frequently recorded in the early modern era with the specific purpose of exploring and salvaging ship wrecks [27]. Over the centuries later, humans have developed technologies such as submersibles, bathyspheres [116], and autonomous underwater vehicles (AUVs) [117] which allow longer travel distance, deeper dives, and autonomous underwater navigation (Fig. 1.1).

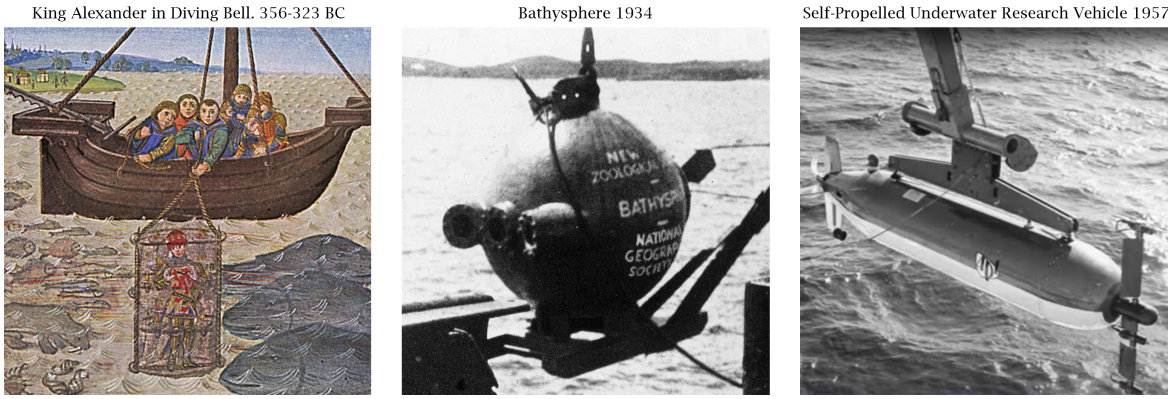


Figure 1.1: Equipments being used in benthic exploration in the history.

By combining the technology of passive deep-sea observation vessels and powered underwater vehicles, one of the first deep-sea research submersibles, the DSV Alvin, was commissioned in 1964 and is in active service today, operated by the WHOI [41]. Equipped with robot arms and camera systems, DSV Alvin has made more than 5,200 dives since commission, including exploring the wreck of RMS Titanic.

Over time, advances in the development of underwater vehicles have led to increased autonomy and improved imaging quality, enabling more effective studies of benthic ecosystems and geography through visual data. The Seabed class AUV was designed and developed in the early 2000s as an unmanned vehicle that could be operated autonomously without a tether to the mother ship [104]. To meet the growing interests in seafloor optical imaging and reconstruction, Seabed AUVs are equipped with high-resolution high dynamic range (HDR) imaging systems together with Doppler Velocity Logger (DVL), Ultra-short baseline acoustic positioning system (USBL), depth sensor, acoustic current profiler and side scan sonars. The dual-hull design passively stabilizes the vehicle under currents and waves, allowing higher quality photographic mapping of the sea floor within a close range, even in rugged terrain and pure dark environment (Fig. 1.2).

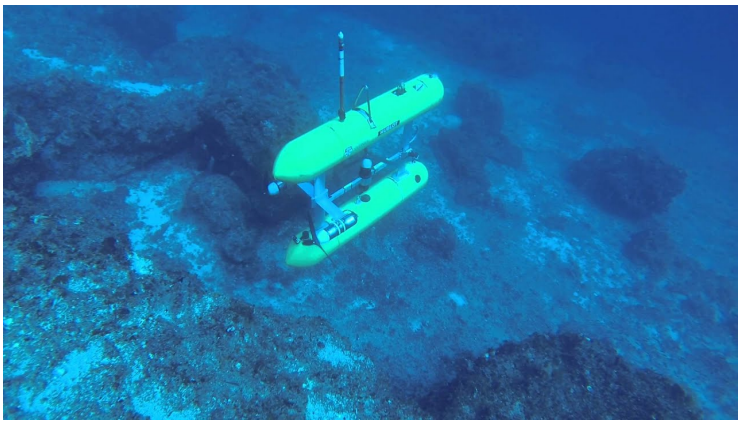


Figure 1.2: Sirius, a Seabed class AUV operated by The Australian Centre for Field Robotics (ACFR), scans the seafloor with color imaging system. A significant portion of data used in chaptor 7 are collected by Sirius AUV during 2009-2015 in the water body around Australia.

Meanwhile, achievements in computer vision and machine learning have also been adapted to underwater robotic images. Researchers from ACFR widely employ AUVs to survey the coral site around Australia, reconstruct the coral bed into a 3D model with RGB image observations, and apply machine learning methods to classify the images and segment the 3D map [108] (Fig. 1.3). This effort helps scientists and the public better understand the deep-sea environment and enabled

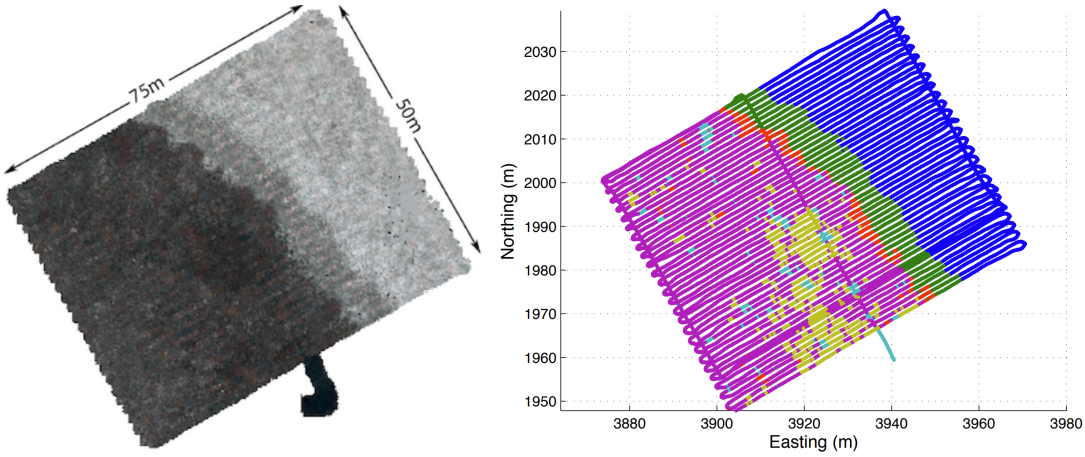
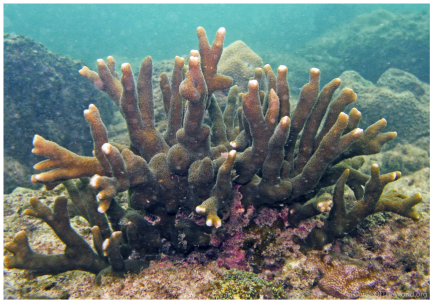


Figure 1.3: A rectangle-shaped area of Scott Reefs (300 km northwest of Australia) mapped with Sirius AUV. Unsupervised machine learning methods are used to segment the map from corals, barren sands, and transition ares as shown in the plot in the right [108].

the study of the temporal change of habitat in certain areas over time.

Although visual mapping technologies matured over time, challenges remain in underwater imaging. Due to limited lighting underwater and light-water interactions, images taken by an underwater robot always exhibit perturbations in appearances, such as color distortion, refraction, water caustics, dynamic and unevenly distributed illumination in dark environments, etc. Those distinct challenges underwater hinder human understanding of marine biology and ecology. For example, different coral species can exhibit similar 3D structures. In some of the cases, colors can be used as a cue to tell them apart. An example is shown in Figure 1.4, that two different coral species, Montipora Digitata and Montipora Samarensis, present similar structures but subtle color differences. However, lighting effected mentioned above will distort the color and texture presented in the images, challenging the accurate human analysis from the observations. Therefore, removing lighting effects that perturb observed colors is of great interests to the community.

In this dissertation, I will discuss a series of problems unique to underwater robot perception under lighting perturbations, and how they are tackled by building 3D representations, including water caustic removal, color correction and synthesis, and dark-environment relighting. Then I will discuss how photorealistic 3D representations of underwater terrain can be generated by deep generative models. I will introduce algorithms that combine geometry and optic models with modern deep learning and 3D computer vision approaches that address these problems on real-world robotic data, with zero annotations.



Montipora Digitata
(Pale brown body and white tip)

Montipora Samarensis
(Pale cream/brown, pink, blue)

Figure 1.4: The corals as shown in this figure, Montipora Digitata and Montipora Samarensis, belong to same genus but different species. They often present similar structure but subtle color difference. Removing lighting effects in the images can help recovering color information and discerning one specie from another.

1.2 Problem statements and contributions

This section highlights the problem formulations and main contributions in this dissertation. We start with the problem of water caustic removal in shallow water where illumination is dominated by natural light. We then talk about how to remove color distortion in underwater images and how to relight the dark underwater environment, the setup of both problems are inspired by deep water robot deployments where major illumination source is attached to the robot. Finally, we will discuss underwater 3D terrain generation with deep generative models. Together, we show that underwater perception problems can be framed as learning 3D representations in concert with physical laws, stochastic processes, and foundation models, in the absence of labels, annotations, captions or any form of human supervision.

How to remove caustic effects on the seafloor with natural illuminations?

For underwater robots deployed in shallow water, which is mainly illuminated by natural light, images taken often present heavy water caustics. This is because of the natural light refracted by the dynamic wave surface. In some literatures, water caustics is also addressed as sunlight flickering. Water caustics can downgrade the 3D reconstruction quality due to photometric inconsistency. Can we build photorealistic models that render the scene caustic free?

In chapter 3 we propose RecGS [130], which is a pipeline that recurrently builds 3D Gaussian Splatting (3DGS) models of the seafloor and removes the caustics at the same time. We observe

that the caustic pattern is often separable in the frequency domain, and we achieve this with 2D Fast Fourier Transform (FFT). In each recurrence, the residuals of the rendered image from the 3DGS model are decomposed with FFT, and the low-frequency part of the FFT is considered caustics and truncated from the residual. Experiments show that, within 40 iterations, the cuastic level in the 3D representation is reduced to a minimum level. We collected data from real-world marine environment and show that our approach effectively removes caustic effects with both quantitative and qualitative evidence.

How to restore color from multi-view underwater observations?

Coloration of marine species and substrates is widely used in studying marine biology, geography, and archaeology, etc. For example, the color of the coral reef can be used to examine the health condition of coral. However, water effects, such as attenuation and backscattering, often distort coloration in underwater images. Existing methods apply classic image processing methods or neural networks on a single image, making it perform poorly on novel observations. Given the image sequence taken by robots in the deep sea, can we recover the true coloration from this multi-view observations?

In Chapter 4, we adapt neural reflectance field [15], a variant of neural radiace field (NeRF) [67], to deep water imaging. By integrating the physical model of light-water interaction and a neural reflectance field, we build a model that learns water effects and 3D geometry of the scene at the same time. Therefore we can reconstruct the scene without water effects, and synthesize the scene with novel water effects from novel views, with photorealistic rendering quality. We show that our approach out-performs baselines on both synthetic and real-world data.

How to build 3D photorealistic map with onboard lightsource

For robots exploring dark environments, such as deep sea or caves, they often carry cameras and an onboard light source moving as a rigid body. With this moving light source setup, it is hard to reconstruct the environment with photorealistic models such as NeRF [67] or 3DGS [49] since they are all optimized with photometric loss. How to model such a camera-light setup and reconstruct and relight the environment in darkness?

In Chapter 5 we propose neural light simulator (NeLiS), which learns the model of a camera-light source system. With a sequence of calibration images taken from multiple views, the intrinsic and extrinsic parameters of this model are optimized. With this model, we are able to simulate the

light cone with photorealistic quality. In Chapter 6, we propose DarkGS, a variant of 3DGS [49], which reconstructs the environment as 3D Gaussians with a sequence of images lit with an onboard light source. When building the 3D Gaussian representation, NeLiS allows ray tracing from the camera to the scene, then back to the light source. The DarkGS model can be then relighted by virtual light sources, such as a parallel light source, to present the appearance of the environment under normal lighting. Visualization and numerical analysis in both calibration and deployment environment show great effectiveness of our proposed approach.

How to synthesize photorealistic underwater 3D scenes?

3D assets such as the reconstruction of environments can also be used to simulate robots by providing realistic virtual spaces where robotic systems can be tested and trained. Diverse simulation environments help robot algorithms to generalize by exposing them to a wide range of scenarios and edge cases. While a large amount of underwater data captured by robots is publicly available, only a small portion of it is suitable for per-scene 3D reconstruction. Given the large amount of underwater image dataset collected from underwater robot deployments, can we train a deep generative model that generates diverse 3D underwater scenes?

In Chapter 7, we propose DreamSea, a deep generative model that generates realistic underwater scene based on diffusion models. The neural network in DreamSea is a U-Net architecture [94] trained on underwater robotic images. The images are generated under the control of fractal latents, which preserve the natural variation of the appearance. Images are stitched and refined by 2D diffusion priors to a 3DGS [49] representation. Extensive experiments show that our approach is effective in generating large-scale, diverse, and realistic underwater terrains that present natural variation.

1.3 Acronyms

Acronym	Full Name
AUV	Autonomous Underwater Vehicle
CNN	Convolutional Neural Network
DVL	Doppler Velocity Log
fBm	fractal Brownian motion
FFT	Fast Fourier Transform
HDR	High Dynamic Range
MLP	Multilayer Perceptron
MSE	Mean Squared Error
NeLiS	Neural Light Simulator
NeRF	Neural Radiance Field
RID	Radiant Intensity Distribution
ROV	Remotely Operated (Underwater) Vehicle
RTE	Radiative Transfer Equation
SDS	Score Distillation Sampling
SfM	Structure from Motion
SLAM	Simultaneous Localization and Mapping
TAN	Terrain-Aided Navigation
UAV	Unmanned Autonomous Vehicle
USBL	Ultra-Short Baseline acoustic positioning system
VFM	Visual Foundation Models
VLM	Visual Language Models
VRE	Volume Rendering Equation
VSF	Volume Scattering Function
3DGS	3D Gaussian Splatting

1.4 Related Publications & Presentations

Beyond NeRF Underwater: Learning Neural Reflectance Fields for True Color Correction of Marine Imagery (Chapter 3)
Authors: **Tianyi Zhang** and Matthew Johnson-Roberson
Venues: RA-L 2023; ICRA 2024

Learning Neural Reflectance Fields for True Color Correction and Novel-View Synthesis of Underwater Robotic Imagery (Chapter 3)
Authors: **Tianyi Zhang**, Qilin Sun and Matthew Johnson-Roberson
Venues: IROS 2023 PIES Workshop

DarkGS: Learning Neural Illumination and 3D Gaussians Relighting for Robotic Exploration in the Dark (Chapter 4, 5)
Authors: **Tianyi Zhang**, Kaining Huang, Weiming Zhi and Matthew Johnson-Roberson
Venues: IROS 2024 (Oral); ICRA 2024 RoboNeRF Workshop

RecGS: Removing Water Caustic with Recurrent Gaussian Splatting (Chapter 6)
Authors: **Tianyi Zhang**, Weiming Zhi, Kaining Huang, Joshua Mangelson, Corina Barbalata and Matthew Johnson-Roberson
Venues: RA-L 2025; ICRA 2025

Infinite Leagues Under the Sea: Photorealistic 3D Underwater Terrain Generation by Latent Fractal Diffusion Models (Chapter 7)
Authors: **Tianyi Zhang**, Weiming Zhi, Joshua Mangelson and Matthew Johnson-Roberson
Venues: ICLR FM-Wild Workshop (Oral); Under Review

Chapter 2

Background

Robots perceive their 3D environments through on-board sensors, among which RGB cameras are the most commonly used on various kinds of robot platform. Cameras project the appearance of a 3D environment onto a 2D image. This dissertation studies how to infer and generate 3D information with RGB images in an underwater environment, with dynamic and complex lighting effects. However, although there are various sensors such as LiDARs and sonars which acquire 3D environmental information directly, they are not as commonly used as cameras due to downgraded performance underwater and high costs; therefore, they are not discussed in this dissertation.

This chapter will introduce some basic concepts and technologies that serve as the foundation of this dissertation, including 3D reconstruction, photorealistic 3D representations, and generative models.

2.1 3D Reconstruction from Images

3D reconstruction of an underwater scene from RGB image observations is one of the most expressive way to convey information about an underwater scene. In this section, we will introduce some generic concepts of 3D reconstruction from a minimal setup, 2-view stereo vision, to multi-view and real-time incremental reconstruction, i.e. structure from motion (SfM) and simultaneous localization and mapping (SLAM), which are based on optimization frameworks. We will also briefly discuss the emerging deep-learning based feed-forward 3D reconstruction methods.

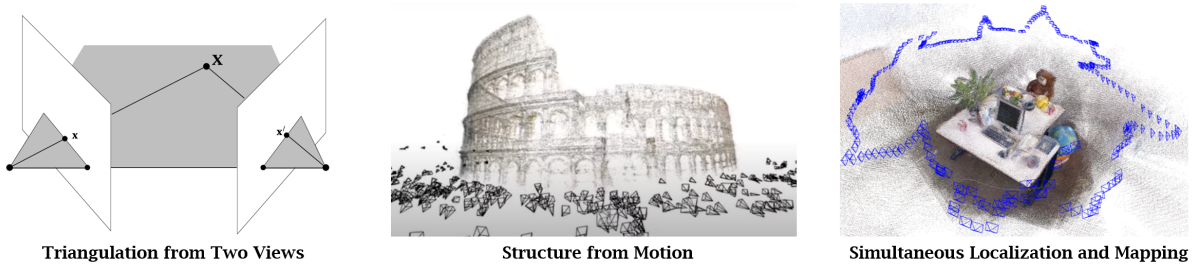


Figure 2.1: Illustrations of triangulation from two-view images [38], SfM [3], and SLAM [20].

2.1.1 Stereo Vision

To recover 3D information from 2D images, early methods leverage cues from multiple view observations. A base example is a stereo camera, which is a camera system consists of 2 cameras. The extrinsic between both cameras, that is, the rotation and translation between the coordinate frames of both cameras, can be estimated with the camera calibration pipelines [90]. Feature point extraction and matching methods [55, 56] find the corresponded points in both images, and estimate the intersection of lines of sights as the 3D coordinate of the point. This process is also known as triangulation. By triangulating all the corresponded 2D points in both images, a 3D point cloud can be extracted which describes the 3D structure of the scene.

2.1.2 Structure from Motion (SfM)

The same principle of 2-view stereo can be applied to multiple views. Instead of triangulation, the position of 3D points can be optimized by minimize reprojection error on 2D images. The camera poses are optimized together with 3D points as well. Such techniques are generally referred to as Structure from Motion (SfM). The theoretical foundation for multi-view geometry in SfM is laid by Hartley and Zisserman [38]. Agarwal et al. [3] applied SfM to city scale modeling from unorganized web photos, which possess different unknown camera parameters and illumination. SfM technology is later developed into open source softwares such as COLMAP [100], opensfm and OpenMVS, as well as commercial softwares.

2.1.3 Simultaneous Localization and Mapping (SLAM)

Robotic applications often need to estimate the poses of the robot and map the environment incrementally in real time. Here, SfM which solves the global optimization problem iteratively, is

not suitable for deployment on robots. Instead, the problem is studied within the scope of SLAM. Early SLAM technologies are based on filtering algorithms, such as the Kalman filter [18], particle filter [48] and their variants. Modern SLAM approaches often build the observations into a factor graph, and optimize the 3D map and camera poses in a sliding window [20], with loop closure mechanisms to correct the accumulated error.

In this dissertation (ch. 3), while our data are collected by mobile robots, they are processed offline. So we use SfM solutions to acquire the point cloud map of the environment and the camera poses.

2.1.4 Feed-forward methods

Deep learning approaches are widely used in monocular depth estimation and sparse-view 3D prediction, where geometry correspondence are sparse. Monocular depth estimation is the task of predicting the scene depth (per pixel distance) of objects in a scene using only one RGB image. Unlike stereo vision (which relies on geometry), monocular depth estimation is ill-posed: from one image alone, multiple 3D scenes could result in the same 2D picture. Therefore, learning-based methods are often used to estimate depth based on prior knowledge. Early work trained popular network architectures such as CNN [54] and transformers [88] on annotated datasets such as NYUv2 [103] and KITTI [34]. Recent work [122] leverages internet-scale unannotated data as well as human supervision in the loop, so that the neural network generalizes well on unseen data. Such models are also considered depth foundation models. In ch. 7, we distill DepthAnything v2 model [122] to train a diffusion model that generates benthic images with realistic appearance and depth.

Deep neural networks such as transformers and diffusion models can also be used to predict the 3D structure from sparse views [113, 125]. These models are being trained on 3D datasets with ground-truth camera poses and parameters, e.g. CO3D [91], ScanNet [22] and ARKit [11]. Although these methods are not used in this dissertation, they can be the potential substitution for SfM algorithms for dynamic and challenging environments in the future.

2.2 Photorealistic Scene Representations

Photorealistic scene representations refer to the 3D scene structure can be rendered from arbitrary novel view with photorealistic quality. Within the context of this dissertation, photorealistic

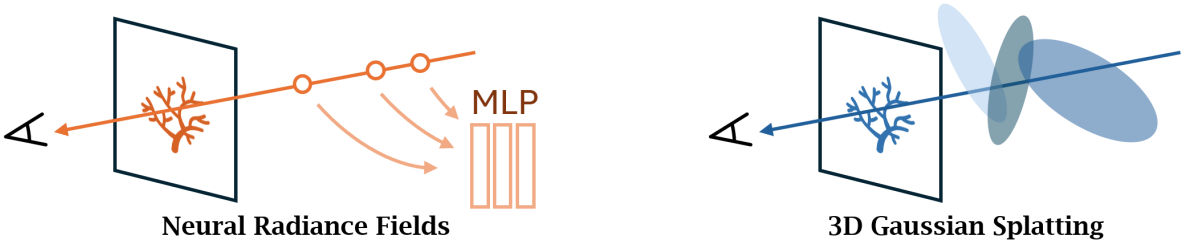


Figure 2.2: NeRF [67] samples along each camera rays and query the neural network for each sample points. 3DGS [49] projects the 3D Gaussians onto 2D image plane, which can be parallelized and does not require any sampling.

rendering of seafloor can help scientists better understart the observations from novel views, as well as helping roboticist better simulate the environment.

Traditional 3D scene structures such as voxels, point clouds, or meshes are not considered photorealistic because of their discrete nature. In this thesis, we will mainly talk about NeRF (Neural Radiance Field) and 3D Gaussian Splatting (3DGS), and their variants.

2.2.1 Neural Radiance Fields (NeRF)

NeRF [66] is a neural implicit representation that learns a 3D scene in the form of a neural field of volume density and radiance. NeRF is usually realized with an MLP, the input of which is positional embeddings of 3D coordinates being queryed, and the output is the RGB values and opacity value. By taking the camera poses estimated from an SfM pipeline, volume rendering equations (VRE) can be applied to render an image from a NeRF representation. By minimizing the error between the ground truth image and rendered image, the weight of the neural network are optimized to produce coherent 3D appearance under diverse views.

The VRE in NeRF, which are based on Radiative Transfer Equation (RTE), are not only good for inferring the 3D geometry of objects but also have the power to model the effects of water such as absorption and scattering. Based on NeRF's framework, neural reflectance field [14] and its variants [112, 123] model the reflectance of the scene, which enables high-quality rendering under novel lighting conditions.

For underwater scenes illuminated by light sources attached to the robot, the appearance of the scene changes due to the robot's movement. To accommodate for these appearance changes resulting from varying illumination conditions, it is necessary to model reflectance properties of the scene. Therefore, we opt to use a neural reflectance field [14] backboned by iNGP [73] as our

foundational model.

2.2.2 3D Gaussian Splatting (3DGS)

Although popular variations of NeRF such as Mip-NeRF[9] and Instant-NGP [73] have improved NeRF by rendering accuracy and speed, the efficiency of the algorithm is still heavily bottlenecked by the sampling cost. This has become a common issue for all kinds of implicit methods, which require a large amount of sampling to get high-quality rendering results.

Recent innovation introduced as 3DGS [49] has revolutionized this field by replacing the neural network backbone with spatial Gaussians, allowing rendering speed of 100+ fps while maintaining the accuracy and differentiability of the system. 3DGS is based on the same radiance assumption with NeRF, that models each point in the scene with emissive radiance. A scene is typically represented by a set of 3D Gaussian distributions. The exact number of Gaussians depends on the size of the scene and image resolution. For each image view in the training set, 3D Gaussians are projected and sorted, to generate the color value for each pixel. Since the 3D Gaussian representation does not require sampling, the rendering operation can be highly vectorized and parallelized, achieving high efficiency rendering.

For our robotic setup, the problem of varying illumination is attempted by [15, 127] which models the physical property of the scene, e.g. roughness, albedo, and normal, instead of the radiance as a constant. However, these works oversimplify the light model as a Lambertian point light source, co-centered with the camera, making it not applicable on many real robotic platforms. Gaussianshader [45] and Relightable 3DG [33] introduce physical properties into the 3DGS framework but model illumination as a constant that does not change frame-by-frame. According to our experiments, none of the above mentioned methods handles the illumination-inconsistency issue on a real-robot imaging system.

2.3 Generative Models

This section will introduce deep generative models that can be used to create diverse and realistic visual contents, which consequently lead to the contribution of benthic terrain generation in this dissertation.

2.3.1 2D Generative Models

Given an image dataset, an image generation model learns the distribution of this dataset. Unseen image samples can be generated as samples drawn from this distribution. Early techniques such as Variational Autoencoders (VAEs) [51] and Generative Adversarial Networks (GANs) [35] are able to generate realistic images. In recent years, models such as DDPM [40], Stable Diffusion [93] and DiT [77] allow high-quality generation that can be conditioned on language inputs. These technologies have also led to commercialized models such as ChatGPT and SORA (OpenAI's Video Generative Model). Although these models are capable of creating arbitrary scenes, we find, empirically, that the quality of generated underwater scenes is significantly lower than that of other more common environments. It can be hypothesized that the training data for underwater scenes is scarce and unbalanced. The development of specialized models with curated data for underwater scenes is still an open problem. In Chapter 7, our proposed DreamSea model uses a DDPM [40] network with the RePaint [57] framework as a backbone image generation and inpainting model.

2.3.2 3D Generative Models

Existing models that generate 3D assets or scenes can be categorized into two types: 1) Models that directly generate 3D representations in a feed-forward pass; 2) Models that create 3D representations supervised by 2D diffusion priors. The former type requires massive amounts of 3D data and only applies to specific data structures, such as point clouds or voxels. The latter is more flexible in training since only RGB images are needed, which is much easier to acquire than 3D data.

In Chapter 7 of this dissertation, the goal is to develop a model that can generate realistic 3D underwater scenes. However, 3D dataset for underwater environments remains scarce, which hinders the use of models directly being trained on 3D data and output 3D representations. So we resort to the latter approach. According to existing research [82, 109, 124], a 2D diffusion model can be used to supervise a NeRF or 3DGS model. By iteratively optimizing the representation with Score Distilled Sampling (SDS) loss, the 3D model will present coherent and realistic appearance across different views. In Chapter 7, with a trained 2D diffusion model that generates realistic images of seafloor observations, we adopt the method presented in [109, 124] to refine the generated 3D terrain to get better rendering quality.

Chapter 3

Unsupervised Underwater Caustic Removal

3.1 Problem Setup

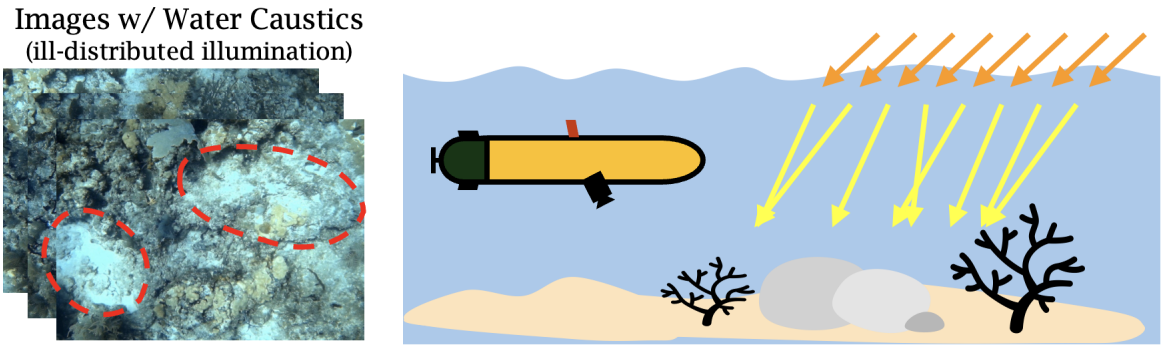


Figure 3.1: Data collected by robots in shallow water is affected by caustics due to refraction from wave surface.

Water caustics, also known as sunlight flickering, is one of many water effects commonly observed. It refers to the phenomenon of shifting patterns on underwater surfaces, caused by light rays that refract through moving wave surfaces (Fig. 3.1). This chapter will focus on developing a 3D vision algorithm that removes the underwater caustic effect from a sequence of *shallow-water* seafloor observations. Here *shallow-water* refers to the underwater area where the main source of light is natural light. We seek to obtain photorealistic rendering of underwater scenes free of water caustics.

Within the literature of 3D reconstruction, SfM methods are some of the most fundamental.

These methods recover the camera poses and sparse 3D structure of the scene from a sequence of images [3]. However, the visualization results of SfM are generally sparse and not photorealistic. Therefore, recent advancements in 3D reconstruction such as NeRF [66] and 3DGS [49] build on top of the results of SfM to enable photorealistic novel-view rendering of the scene. These methods constitute the SOTA of 3D reconstruction. In general, SfM pipelines are robust against illumination change due to sophisticated outlier rejection mechanisms. Therefore, they can be applied to internet data with different levels of ambient lighting and camera parameters [3], while still capable of accurately estimating the poses of the cameras. We also observe that water caustics generally have a limited impact on traditional SfM pipelines for the same reason. Recovering photorealistic visualizations is of particular significance underwater and has implications for other branches of science. However, water caustics are detrimental to the construction of photorealistic representations, which are built on top of SfM. These include NeRF and 3DGS which further learn the 3D scene by optimizing photometric loss instead of re-projection error. As a result, these models are critically affected by the illumination inconsistency from caustics.

Existing studies on caustic removal are based either on image filtering or supervised deep learning. Filtering-based methods [36, 98, 102] estimate and separate the caustic by running a non-linear temporal filter with motion compensation. However, such hand-crafted filtering techniques assume that the motion is small and has no awareness of 3D structure, thus the performance can be easily downgraded due to non-optimal hyperparameters, increased motion speed, and seafloor rugosity. Deep learning methods [5, 31] require an annotated dataset on which to train. However, such dense caustic annotations are extremely expensive to acquire, hard to scale up, and only monochromatic [5]. Therefore, methods based on deep learning perform poorly when generalizing to novel observations.

In this chapter, we study caustic removal as a problem of illumination inconsistency within 3D structures. The community has seen related problems with color [127], brightness [129] and exposure inconsistency [23, 114] being addressed with 3D vision techniques without any annotations. We seek to push the boundaries of this line of work and exploit the 3D consistency maintained in representation to remove caustics.

3.2 Related Works

3.2.1 Classic caustic removal methods

Early work removes underwater caustic with a sequence of consecutive frames taken in a short period. In [98], Spatial derivatives are extracted from each image frame and temporal median filtering is performed to eliminate wide-band caustics. However, the camera motion is not considered in this approach. A method based on non-linear filtering is further developed in [36], which leverages a similar setup that averages out the inconsistent illumination in a sequence of images, but with camera motion compensated. Follow-up work [102] enables online caustic filtering based on a similar principle, but based on the assumption that the camera motion is smooth and the seafloor is flat, hindering the generalizability on real-world images. An adaptive filtering strategy was proposed in [111] to optimize the filter parameters in real time, but as acknowledged by the authors, the performance is still sensitive to hyperparameter tuning.

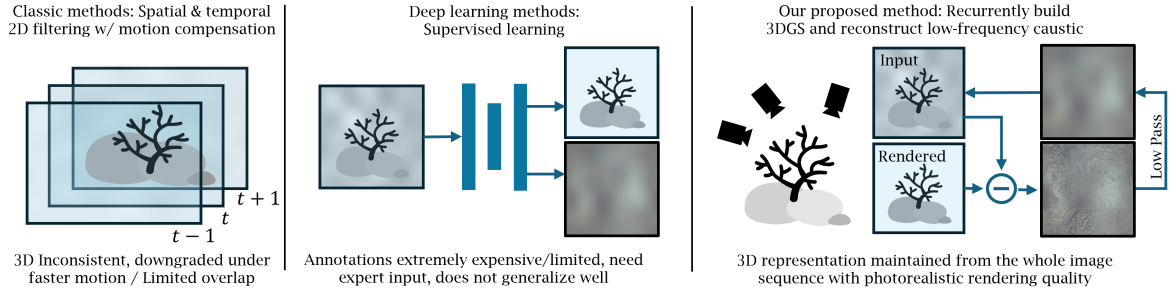


Figure 3.2: Related Works: **Classic methods** (Left) are based on image filtering along a certain time window on the 2D image space. The performance gets downgraded when a 3D structure is present in the scene, or if the camera moves fast and observes less overlap between frames. **Deep learning methods** (Mid) require expert annotations, which is extremely expensive to scale up. Neural networks trained with limited annotated data do not generalize well to novel observations. **Our proposed method** (Right) maintains dense 3D scene representations by building a 3D Gaussian model recurrently and decomposing low-frequency caustics from residuals. Our method works well on images captured from an underwater robot in the wild, without any pretraining on a dataset.

Overall, classic computer vision methods such as pose estimation and nonlinear filtering has been widely employed to compensate caustics in the water, but the operations are all conducted on 2D image space, which has limited capability for handling 3D structured observations and dealing with a fast moving camera with limited overlap from frame to frame. The more 3D structure is

present in the scene, the more the results tend to degrade.

3.2.2 Deep learning-based caustic removal

Deep neural networks have shown great ability to restore images from noise and perturbations [94, 115], with the potential to extend to different modalities and physical phenomena [19]. DeepCaustic [31] proposed a pipeline formed of two CNN-based networks. The first network learns a saliency map, and the second learns to recover the image free of caustics. Stereo and multi-view vision is studied in [5] to remove caustics with a neural net while keeping the seafloor structure invariant in the restored image. However, all deep learning based methods mentioned above need to train the neural network on a dataset with ground truth annotations and masks, making it hard to be scaled up and perform poorly when being generalized to novel observations.

3.3 Methodology

3.3.1 Preliminary

Given a consecutive image sequence (of seafloor observations), camera poses and a 3D sparse point cloud can be estimated from SfM pipeline such as COLMAP [3]. With this 3D keypoint cloud, a 3D Gaussian cloud G can be initialized following the method in vanilla 3DGS [49].

Given camera pose for each image, a camera ray can be modeled for each pixel. An image $\hat{I}$ can be synthesized by applying the following rendering equation to each pixels [49]:

$$\hat{I} = \sum_{i \in \mathcal{N}} c_i \alpha_i \prod_{j=1}^{i-1} (1 - \alpha_j) \tag{3.1}$$

where c_i and α_i are the RGB radiance and volume density of i^{th} Gaussian from $\mathcal{N}$ ordered Gaussians in the 3D Gaussian model G. In other words, Eq. 3.1 projects G into a 2D image. Given a training image I, G is optimized as follows:

$$\underset{G}{\text{minimize}}\ \mathbf{dist}(I, \hat{I}) \tag{3.2}$$

here $\mathbf{dist}()$ is the distance metric. In this study we use a combination of L1 loss and a D-SSIM as our loss function, which is the same with the vanilla 3DGS [49]. Optimizing G refers to optimizing

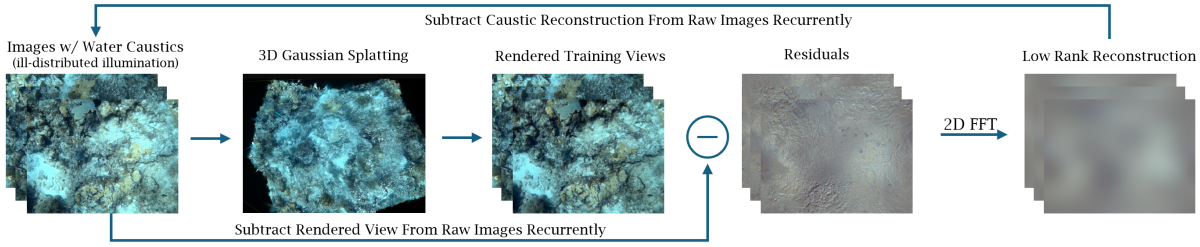


Figure 3.3: Our proposed recurrent 3DGS workflow: we build a vanilla 3DGS model with caustics in the images first, then find the residual between captured image and rendered view. We run 2D FFT on the residual image, and reconstruct it with only the low-rank part. This low-rank reconstruction is then subtracted from the training images.

all the parameters including c_i and α_i of each Gaussian.

Water caustic can be observed in shallow water as the natural light entering the water is refracted at different angles due to the fluctuating wave surface. The refracted light creates focused and defocused pattern on the sea bottom. While it is hard to model the exact physics, we approximate the caustic C as an additive radiance from the scene (similar to previous studies [36]). So the optimization problem becomes:

$$\underset{G}{\text{minimize}}\ \mathbf{dist}(I - C, \hat{I}) \tag{3.3}$$

Note here C has the same image format as the training images, which has all R, G, and B channels. This is different from previous studies which mostly model the caustic as monochromatic [5, 31].

3.3.2 Residual Reconstruction with 2D Fast Fourier Transform (FFT)

From our observation, building a scene with vanilla 3DGS under water caustic will suffer from photometric inconsistency. The caustic pattern, which causes such an inconsistency, is reflected in the residual of the rendered image. We propose to approximate the caustic caused by the refraction from the wave surface with 2D low-rank Fourier series. By building a vanilla 3DGS from the image sequence, the residual can be calculated by $R = I - \hat{I}$. We implemented a low-pass filter with 2D FFT to approximate the low-frequency illumination inconsistency C caused by caustics (this step is similar to the Butterworth filter used in [36]):

$$C = \texttt{ifft}(\texttt{fft}(R)_{[0:k]}) \tag{3.4}$$

Algorithm 1 recurrent 3DGS

Input: Images I, Camera poses
Output: Rendered View $\hat{I}$, Caustic per view C_{ret}
procedure TRAINRECURRENTGS(I)
    minimize $\mathbf{dist}(I, \hat{I})$ with Eq. 3.1
    $C_{ret} = \texttt{zeroslike}(I)$
    while True **do**
        $R = I - \hat{I}$
        $C = \texttt{ifft}(\texttt{fft}(R)_{[0:k]})$ (Eq. 3.4)
        if $\|C - C_{ret}\|_2 < threshold$ **then**
            $C_{ret} = C$
            break
        else
            minimize $\mathbf{dist}(I - C, \hat{I})$ (Eq. 3.3)
            $C_{ret} = C$
        end if
    end while
    Return $\hat{I}$, C_{ret}
end procedure

Here $\texttt{fft}()$ and $\texttt{ifft}()$ are 2D discrete Fourier transform and its inverse operation. $[0:k]$ refers to that only the k frequencies with the lowest absolute value in the frequency space are kept. We use $k = 9$ which is chosen empirically.

3.3.3 Recurrent 3DGS

Since the training data consist of multiple views with dynamic caustics, the 3DGS model, which assumes static lighting, tends to partially average out the caustics. We take advantage of this intuition and propose a recurrent 3DGS framework, RecGS, that progressively removes caustics (see Fig. 3.3 and Algorithm 1).

Starting with a vanilla 3DGS, which is a model for static illumination, the dynamic caustic effects will be overfitted by the 3D Gaussian model at the cost of downgraded rendering quality, for example blurred details. With the FFT-based method introduced in 3.3.2, the low-frequency caustics per frame can be partially separated from the residual. We only remove low-frequency caustics from the training image and then train the 3DGS model again. Since the dynamic caustic is partially removed, the 3DGS model will be less overfitted to caustics and will learn better details

of the scene. We put the above-described process in a loop which will progressively eliminate per-frame caustics and simultaneously improve the rendering quality and consistency of 3DGS model. The convergence of RecGS framework will be reflected by the estimated change of caustics from iteration to iteration. In other words, if the 3DGS model converges, the caustic will remain stable. Therefore, we choose $\|C - C_{ret}\|_2$ as the terminating condition, which is the error between the current and last caustics estimated. If $\|C - C_{ret}\|_2$ falls below a threshold, the model is considered to have converged. See Algorithm 1 for details of RecGS. The algorithm will return the image $\hat{I}$ caustic-free, and caustic C_{ret} which is reconstructed with low rank 2D FFT.

3.4 Experiments

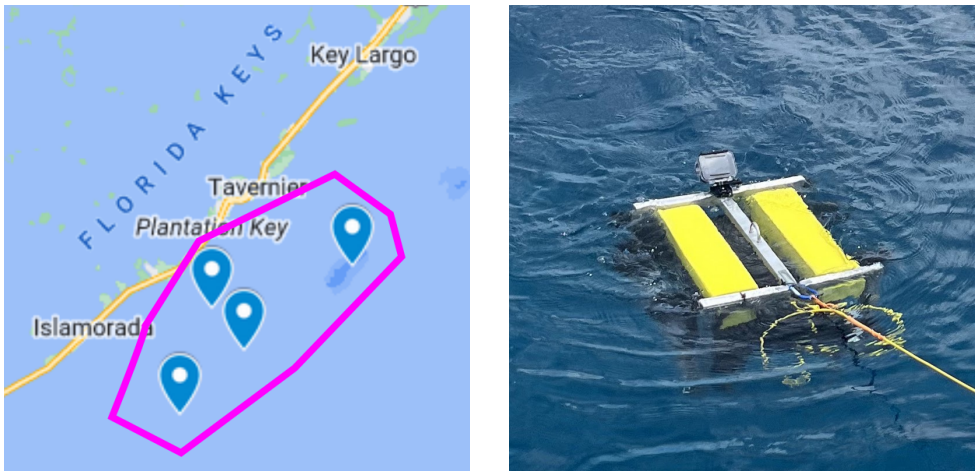
In this section, we provide rigorous empirical evaluations of our proposed RecGS method. We provide details of our experiment setup in section 3.4.1. Then, seek to investigate and provide answers to the following questions:

- Should the caustic effects be learned and optimized jointly with 3DGS? (Section 3.4.2)
- Are pretrained deep-learning approaches sufficient to remove water caustics? (Section 3.4.3)
- How does proposed RecGS compare with alternative filtering-based approaches? (Section 3.4.4)
- How effective are the recurrences, i.e. how many iterations are needed to separate caustics? (Section 3.4.5)

3.4.1 Experiment Setup

Data Collection

We collect a dataset of underwater images by deploying LSU's Bruce ROV [71] equipped with a ZED camera in a coral reef sanctuary in Florida. Illustrations of the location and robot platform are provided in Figure 3.4. The weather during data collection is sunny and the water depth during data collection ranges from 5 to 10 meters.



(a) Data collection site. (b) Robot platform.

Figure 3.4: We collected data in real world marine environment in Florida Keys area. The robot we use is a LSU's Bruce ROV [71] equipped with ZED cameras. The deployments took place in Mid-August, from 10 A.M. to 2 P.M.

Training

Our proposed method trains offline on image sequences collected by the robot. We run COLMAP first on the image sequence to get the camera poses and a sparse point cloud. In our robotic setup, typically a minimum of 30 images are required to build a valid 3DGS model. RecGS is effective on sequences with 30 images or more. We run experiments on 4 selected image sequences with no obvious moving objects and complete COLMAP reconstruction with all the input images. We use the renderer implementation of 3DGS [49] as backbone and PyTorch `fft` module for filtering and caustic reconstruction. Since FFT function consumes negligible computation resources, the runtime of our method is on the same magnitude of vanilla 3DGS and depends on the sequence length and image resolution.

3.4.2 Failure case of joint optimization strategy

The very first question we answer is that instead of decomposing the caustic from residual re-currently, should we jointly optimize per-frame caustic C together with 3DGS? Consider 2D frequency spectrum $\mathbf{c}_{[0:k]}$ which has lowest k frequencies as variables to be optimized and the

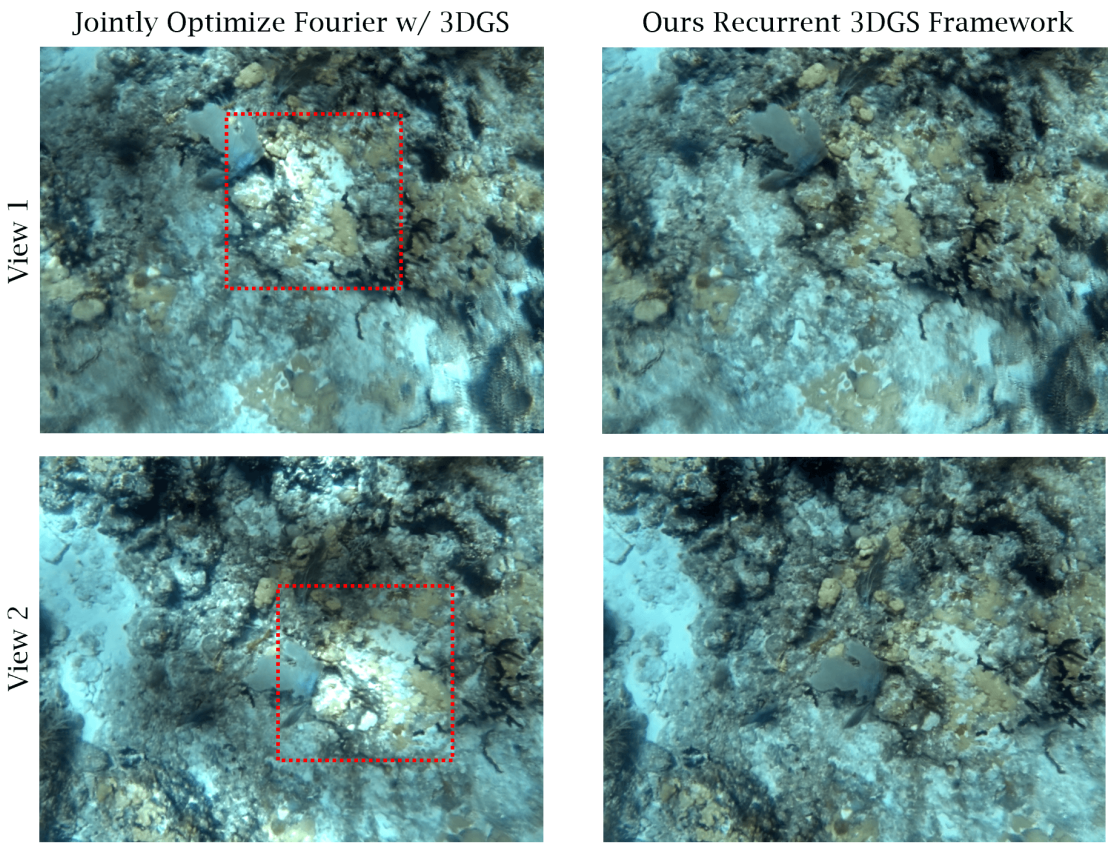


Figure 3.5: Jointly optimizing a low-rank Fourier spectrum $\mathbf{c}$ together with 3DGS leads to an underconstrained behavior. As shown in dashed box, joint-optimization method creates undesired over-exposed areas, while still maintaining multi-view consistency. In comparison, our recurrent method restores the scene with uniform illumination.

high-frequency part are all fixed to 0. Then Eq. 3.4 becomes the following:

$$C = \texttt{ifft}(\mathbf{c}_{[0:k]}) \tag{3.5}$$

And then the optimization problem in Eq. 3.3 will optimize G together with $\mathbf{c}_{[0:k]}$:

$$\underset{G,\mathbf{c}_{[0:k]}}{\text{minimize}}\ \mathbf{dist}(I - C, \hat{I}) \tag{3.6}$$

The results are shown in Fig. 3.5. We can see that jointly optimizing caustic together with 3DGS will lead to ill-balanced illumination. In comparison, our proposed recurrent 3DGS framework behaves constantly to maintain a balanced illumination. We also find that when evaluating the

results of the same scene from different views, both methods can maintain 3D consistency. This implies that while we can learn the caustic model with resort to 3D consistency, such consistency does not guarantee the good performance of the result. We need to carefully engineer the pipeline to constrain the model converging towards the desired direction. Otherwise, as shown in Fig. 3.5, the result of joint-optimization method creates implausible visual results. In this study, our recurrent pipeline shows a great performance gain over its counterpart.

3.4.3 Failure case of Pre-trained Deep Learning Methods

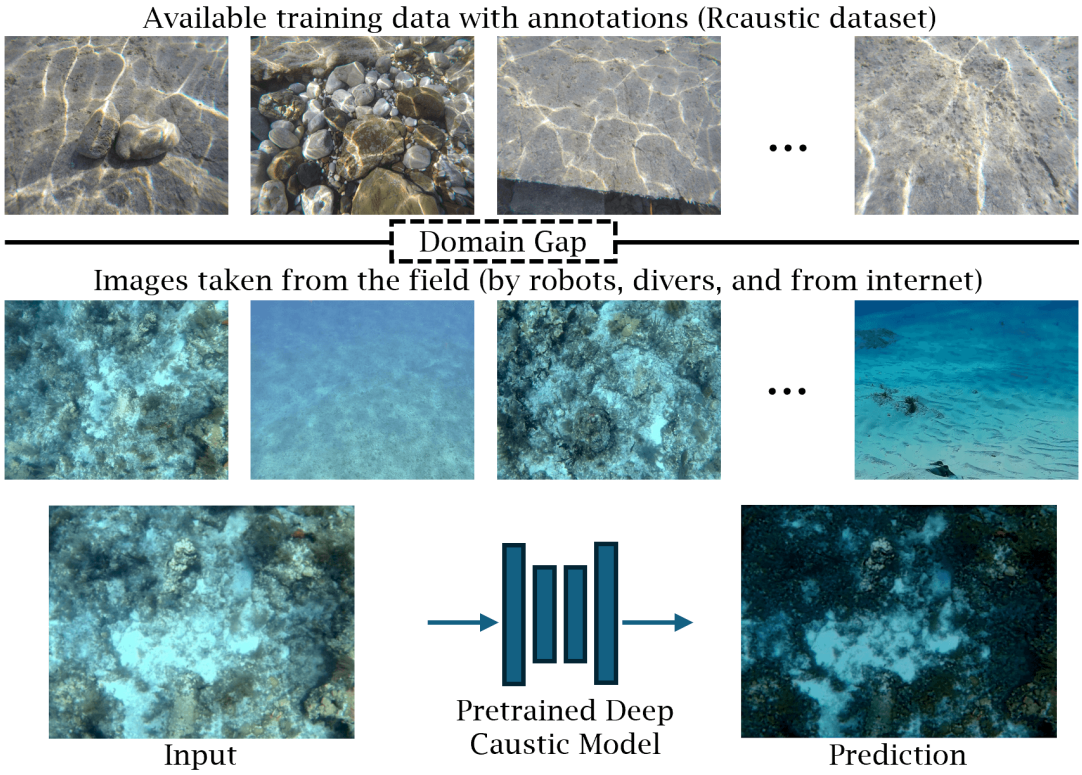


Figure 3.6: Failure of deep learning methods: There is a significant domain gap between the caustic removal training data nowadays and real world data collected from field deployment. Pre-trained deep neural networks can thus perform poorly when such domain gap presents.

The scaling law has played a critical role on a variety of computer vision tasks in the past decade, that with larger amount of data and model size, the model performance gets significantly improved. Deep learning approaches to caustic removal seek to train on a sufficiently large dataset with labeled data and generalize when provided images, under potentially different conditions, and

at new locations. However, the community has seen much less progress in scaling up underwater computer vision studies, as the real-world data are extremely expensive to collect and need expert input to label. The SOTA data for learning caustic is RCaustic [5], which contains 712 pairs/triplets of images from 7 scenes with annotated caustic patterns, including masks, caustic-free ground truth and contours. However, for the monocular camera setup, neural networks trained on such a dataset perform poorly when transferring to novel data collected from the field. In Fig. 3.6, we show that the visual appearance gap between the training data and real-world data, and display an example of the constant failure of pre-trained neural networks when domain gap presents. Our proposed RecGS approach does not suffer the same drawbacks, as it does not rely on training on a labeled dataset to generalize to novel images.

3.4.4 Visualization and Comparison with 2D Filtering Method

The results on different image sequences are visualized in Fig. 3.7. The first column shows the original images as input. We highlight the same area with the most significant caustics from one frame to another in red dashed boxes. The second column shows the results rendered from our recurrent 3DGS framework. By comparing the boxed area in two different frames of the same sequence, we can see that the color and illumination are consistent. The residual column shows the difference between the input image and the rendered image. By performing the low-pass filtering on the residual, we get the caustic result. In comparison, the motion-compensated filtering method (fifth column) performed well on certain sequences, e.g. Sequence 2, but worse than ours on the other sequences, as inconsistent illuminations can still be observed in the boxed area. By comparing the caustic patterns, we also found that while both methods smoothen out the dynamic caustic, the caustic pattern extracted is quite different. We believe that for the 2D filtering method, the background color is more likely to blend with the caustic. The 2D filtering method only uses neighboring images with only one-pass filtering, while our method utilizes all available images and refines the results recurrently.

Table 3.1: MSE↓ of 3DGS on images with caustics removed. All values in the scale of 1e-3.

Train/Val	Seq. 1	Seq. 2	Seq. 3	Seq. 4	Ave.
raw images	4.8/6.7	6.0/10.7	4.2/8.5	3.3/10.2	4.6/9.0
2D filtered	**2.7**/6.1	3.1/7.0	3.0/5.0	1.8/7.7	2.7/6.5
Ours	2.8/**5.4**	**1.4/4.5**	**2.3/3.4**	**1.3/5.9**	**2.0/4.8**

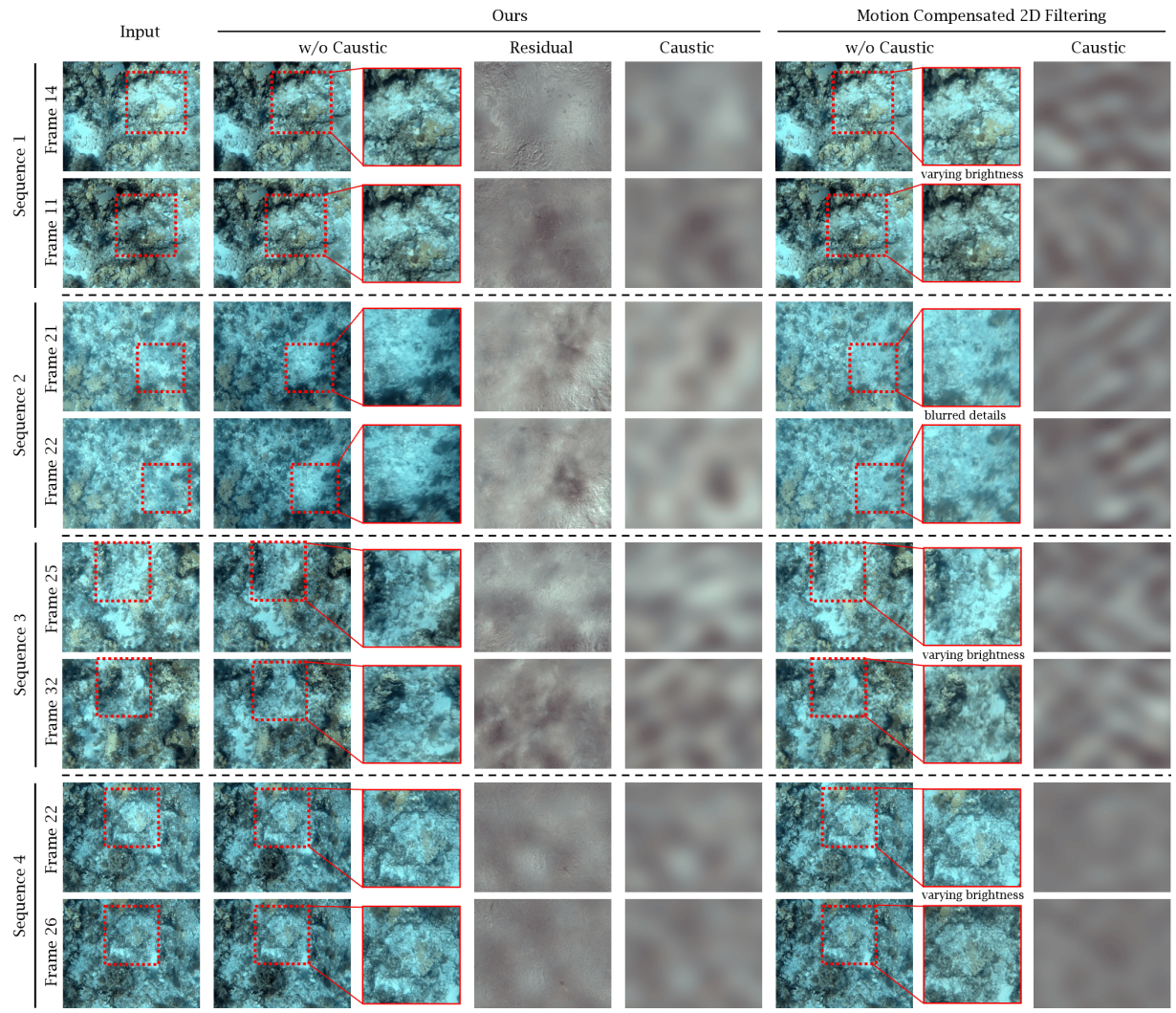


Figure 3.7: Visualization of results from multiple data sequences compared with motion-compensated 2D filtering [36]: For each sequence, we picked up two frames with observable camera motion and caustics between them. We highlight the area with the most significant caustic variation from one frame to another in dashed box. From our results in column 2, we can see that the same areas in different frames are corrected with consistent illumination.

In the absence of ground truth images, we quantitatively evaluate our method by running a vanilla 3DGS model on caustic-removed data, and compare the MSE on training and validation sets. We acknowledge that this metric does not rigorously represent accuracy compared to ground-truth images. However, it reveals the illumination consistency in the caustic-removed images. From Table 7.1 we found that our method on average can reduce MSE on training set by 56% and on

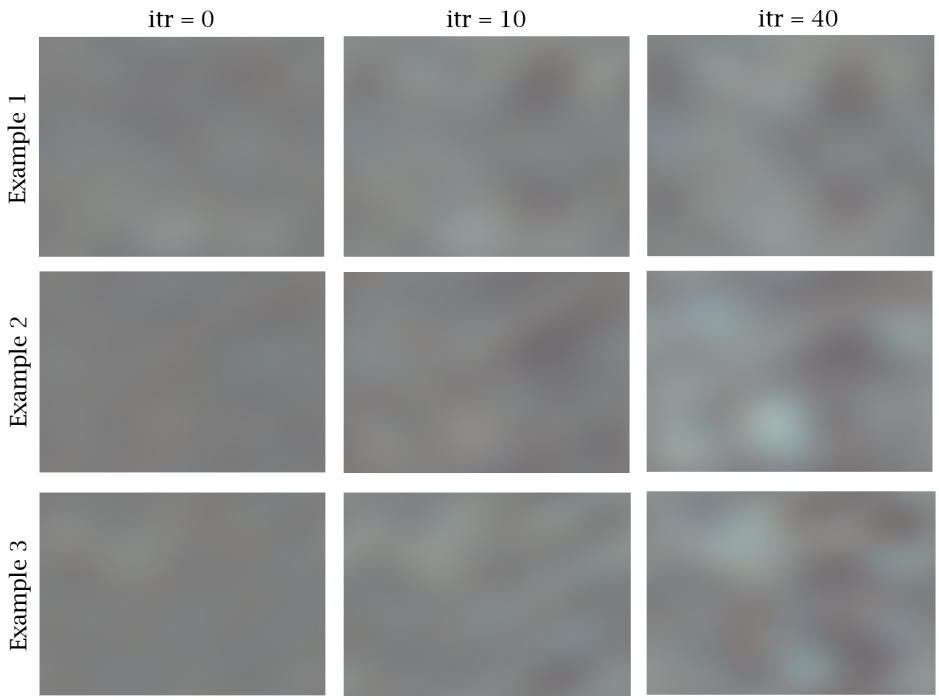


Figure 3.8: Examples from different image sequence: Caustic pattern being learned progressively as the iteration increases.

validation set by 47%, outperforming the motion compensated 2D filtering method's 41% and 28%.

3.4.5 How effective are the recurrences?

According to Algorithm 1, once Eq. 3.2 is optimized with vanilla 3DGS, the 3D Gaussian is considered initialized. Then the algorithm enters the recurrent stage. In each iteration, the caustic is first decomposed from the residual and the 3D Gaussian is optimized for 1000 steps with Eq. 3.3. The way caustics evolve with iterations is shown in Fig. 3.8. In the 0^{th} iteration, that is, trained with only vanilla 3DGS, we can only see a faint imagery of caustic. After 10 iterations, we begin to see a clearer pattern of the caustic. After the model has converged in 40 iterations, we see a clear caustic pattern.

We plot the L_1 error of the caustic returned between two consecutive iterations to support our observation above. A low error means that the model has converged. In Fig. 3.9, we can see an average trend of error declining, especially in the first 5 iterations, and swiftly converging in less than 40 iterations.

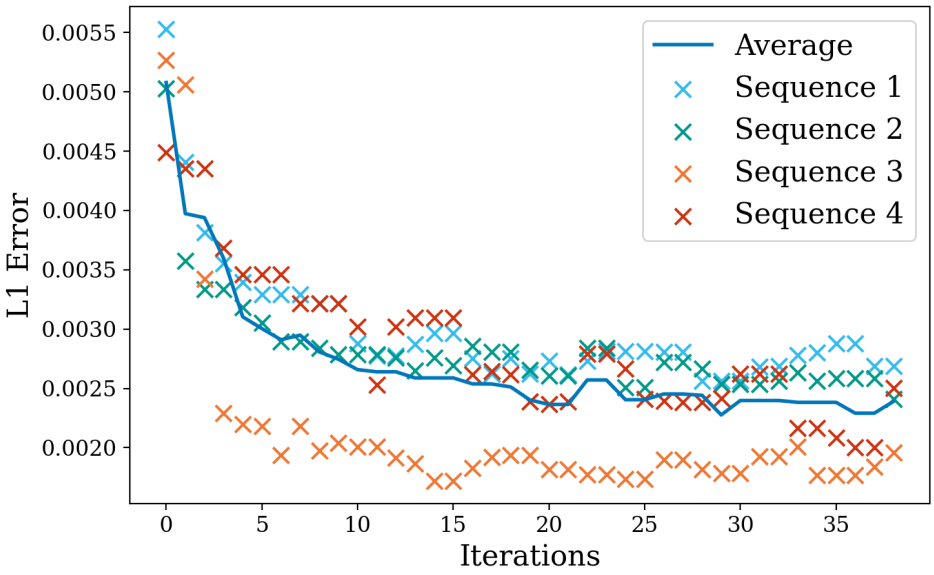
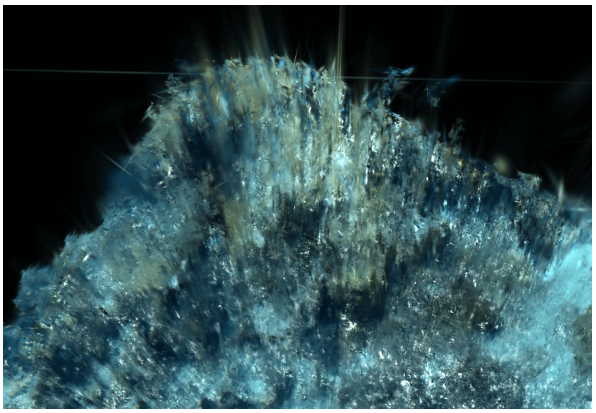


Figure 3.9: Convergence of our proposed recurrent framework. Iterations start after warming up the model with a vanilla 3DGS training pipeline. The caustics are learned progressively.

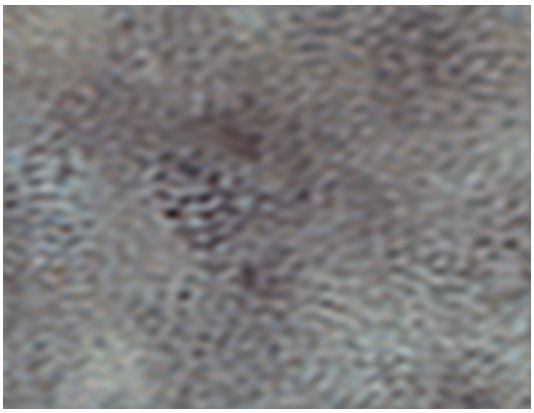
3.5 Limitations

Sparse-View 3D

The 3D structure built with our 3DGS has limited capacity to extrapolate beyond viewing angles provided during training. It can fail when evaluating from a novel side-view, when the training data consists only of top-view images, as shown in Fig. 3.10a. In other words, our method so far works well on top-down views similar to the training views collected with our robot. This limitation is not inherent to our approach, and can be generally observed in NeRF and 3DGS-based methods. In this chapter, we are not able to draw the conclusion that removing caustic helps build a better 3D representation for novel-view photorealistic rendering. The key reason can be that the top-down observations from a robotic setup is not sufficient to constrain 3D geometry. When more advanced novel-view synthesis methods that improve generalization beyond the training images emerge, we can integrate them seamlessly into our recurrent framework and also achieve photorealistic underwater reconstruction.



(a) Gaussians build with sparse top-down views fail when viewing from novel side-views.



(b) Undesired caustics due to improperly selected hyper-parameter.

Figure 3.10: Limitations of our method.

Parameter Tuning

The frequency threshold in the low pass filtering with FFT depends on hand tuning. Choosing an improperly high threshold leads to poor visual results as shown in Fig. 3.10b. Unknown of the proper frequency range can lead to potential failure.

High frequency caustics

We see a wide variety of shallow water images from open dataset and internet present high-frequency and high-intensity caustics. As shown in the first row of Fig. 3.6. Such pattern is rarely observed in our robotic deployment in open water with lower-frequency sea waves. We believe that in the presence of challenging caustic patterns, our recurrent pipeline still makes sense, leveraging the nature of 3DGS that maintains a consistent representation. However, FFT might not be sufficient to model these patterns. Other basis functions or small neural networks that extract the features of such patterns can be considered.

3.6 Conclusion

This chapter proposes a framework that recurrently removes caustic effects from underwater images. 3DGS and 2D low-pass filtering are employed in each iteration to build a illumination-consistent 3D representation of the scene and remove caustic from the residual. The experiments are carried

out in comparison to different strategies such as joint-optimization, deep learning, and 2D filtering approaches, and the results show that our method provides better visual results. Overall, from this project, we learn that by designing a system that wraps around 3D scene representations, we can learn and recover complex illumination effects with 3D visual consistency. Future work includes building better 3D reconstruction from sparse views that allows not only caustic removal, but also improves novel view rendering. In addition, we hope that our proposed recurrent 3DGS can inspire and be applied to different applications with challenging imaging effects.

Chapter 4

Neural Reflectance Field for Underwater Color Correction

4.1 Problem Setup

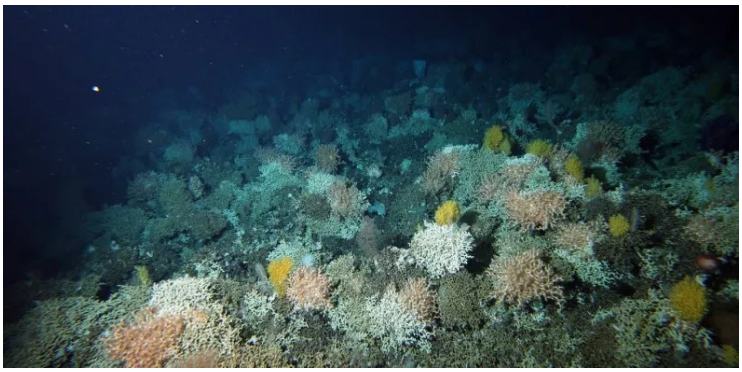


Figure 4.1: Coloration provides important information on evaluating the health of marine organisms such as corals. As shown in deep sea environment, color is distorted due to attenuation and scattering effects in the water. The farther an object is from the camera and light source, the stronger the blue tint and veiling effect become.

The visual information presented in raw RGB format reveals rich details about underwater ecosystems and artifacts. For example, images collected by an underwater robot can be used to assess the health of coral reefs and segment live corals from dead samples [62]. However, the colors displayed in underwater images are consistently distorted due to wavelength-dependent

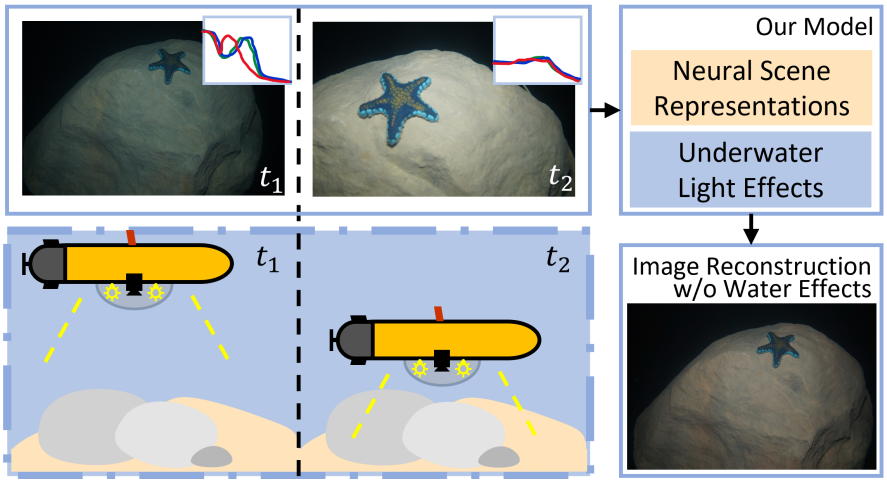


Figure 4.2: Observing an underwater scene from different altitudes results in variating color distribution over the RGB channels. Such observations encode the physics of light-water interactions. Our proposed model leverages this cue to restore the true color of underwater scenes by learning water effects together with neural scene representations.

attenuation and veiling effects resulting from light-water interactions (Fig. 4.1). Such effects alter the visual appearance of images, as well as the performance of downstream tasks such as detection, classification, or segmentation [42]. Restoring the color in underwater imagery is of great interest to communities working on marine ecology, biology, and geography, etc.

The formation of underwater color distortion has seen significant work, in which two kinds of light-water interaction are commonly studied: attenuation and scattering [43, 70]. Attenuation describes the process whereby water absorbs light at varying rates depending on the wavelength. Red light is absorbed most quickly, leading to loss of the red part of the visual spectrum in typical underwater images [83]. Underwater light scattering refers to the process by which light is dispersed in various directions as it interacts with water molecules, suspended particles, and other microscopic elements within the underwater environment [43]. While in graphics multiple-scattering is typically modeled, in water photons reflected to the camera without striking the scene, i.e. backscatter, have a major impact on image formation by creating a veiling effect. Although our understanding of water optics has advanced, restoring color in underwater images is still challenging. While these effects are well-modeled, accurately estimating them from real data in uncontrolled environments remains an open problem.

Early studies on marine optics developed underwater image formation models [43, 65] and measured absorption and scattering functions from different types of water samples [79, 83].

With the above work, images can be synthesized with underwater effects [106]. However, this approach is insufficient for accurately correcting the color of real-world underwater images, as the measurements of a finite number of water optical properties cannot be reliably applied to novel field data. Recent progresses on SfM and deep learning have inspired the development of data-driven algorithms for underwater color correction. SfM-based method [16] estimates the true color (albedo) with multiple-view geometry constraints, but is only able to generate sparse results on feature points. Deep-learning-based methods [42, 52, 101] are able to correct the color with physical cues, but the result depends on prior color distributions or pre-training.

Combining insights from both types of methods, we developed a unified model that effectively restores the true color in underwater imagery (Fig. 4.2). Our proposed model optimizes the attenuation and backscatter coefficients together with a neural reflectance field [15] from a sequence of observations without any assumptions on prior color distributions. Based on the observation that water and scene are separable given volume density, we embed a logistic regression function in our neural scene representation which allows us to apply different light-transmitting physics to water and the scene, while maintaining end-to-end differentiability of our model. Our experiments demonstrate that our method is able to generate photorealistic results with restored true color in a dense format. Unlike previous studies that attempt to correct the color in the underwater images without taking lighting conditions into account, our work build physical model for underwater robots which have onboard light sources and cameras moving as a rigid body, thereby outperforming previous studies.

4.2 Related Works

Underwater Image Formation Model

According to the Jaffe-McGlamery model [43, 65], the formation of underwater images can be decomposed into direct signals, forward-scattering, and backscatter. Direct signals refer to the light that is reflected from the underwater scene. Backscatter refers to the phenomenon in which light enters a camera without being reflected directly from the scene. The trajectory of a photon after interacting with a particle in water is characterized by VSF [79]. These empirical functions are dependent on both viewing and lighting directions. Forward-scattering occurs when a photon deviates from its direct path before reaching the sensor, resulting in a blurred image. This effect can be approximated by convolution operations [43] or Gaussian blurring [106].

In this work, to overcome the challenge of estimating VSF with limited constraints from observations, we propose several approximations applicable to underwater robots. Our scene representations do not model forward scattering, the error introduced by which is zero-mean and negligible [68].

Underwater Color Correction

Early studies on underwater color correction make assumptions on underlying color distributions, e.g. histogram equalization [81], grayworld [17], or dark-channel prior [39]. However, color balanced from the above assumptions lacks consistency when the same scene is observed from multiple views due to range-dependent water effects.

Bryson et al. [16] leverages the physical constraints from multiple-view geometry to estimate the true color of the scene. However, this method only estimates the true color of feature points and is unable to directly generate color-corrected images in a dense format.

Further progress in this field is made with deep learning approaches. WaterGAN [52] proposes to synthesize a dataset with ground truth depth and colors by training a GAN, then train a color correction network to restore the color. FUnIE-GAN [42] emphasizes image quality for downstream tasks rather than adhering to physical constraints and, as such, can achieve real-time performance. GAN-based methods, such as those mentioned above, require pre-training on a dataset. These methods can exhibit biases if the underlying color distribution differs from that of the training set. In contrast, our approach does not require any pre-training on datasets. Rather, it restores color by creating scene representations using a series of observations from multiple perspectives.

WaterNeRF [101] utilizes mip-NeRF [10] to model the underwater scene. Based on depth estimation from mip-NeRF, WaterNeRF learns the absorption and backscatter coefficients by optimizing the Sinkhorn loss between rendered image and histogram equalized image. Our approach diverges from WaterNeRF in that we model the scene as a reflectance field, which accounts for changes in illuminance, as opposed to a radiance field. Furthermore, we do not make any assumptions regarding the underlying color distributions.

Lastly, all the approaches mentioned above [16, 52, 101] use the model proposed in [99] to account for backscatter, which assumes natural and ambient light to be the major illumination source of scattering. In other words, their formulations are based on the assumption that the intensity of scattering is spatially constant, which does not hold for underwater robots equipped with light sources, taking light fall-off into consideration. In our work, we depart from [99]'s

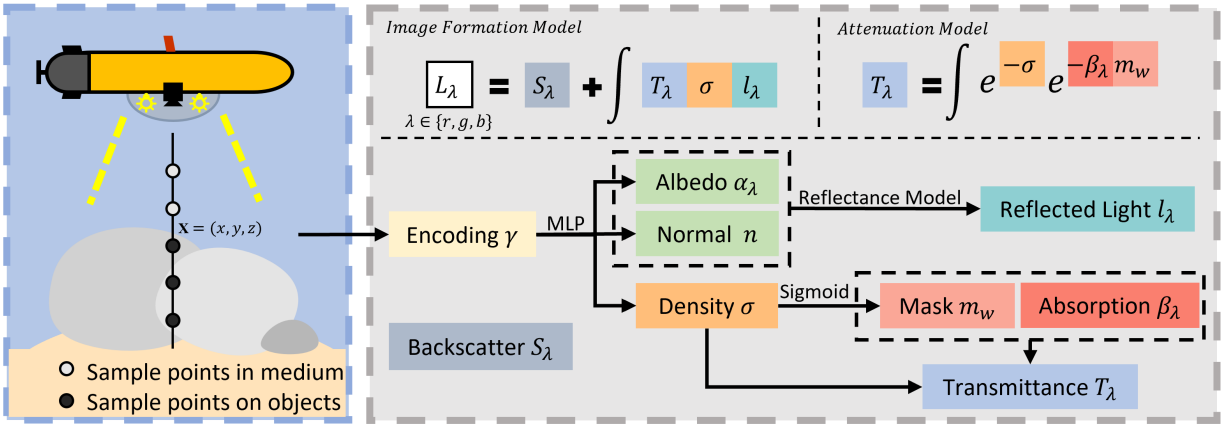


Figure 4.3: Our proposed model: Sample points $\mathbf{x}$ are first mapped into positional encoding $\gamma(\mathbf{x})$, as the input of an MLP. The output of the MLP consists of albedo $\boldsymbol{\alpha}$, surface normal $\mathbf{n}$, and volume density σ. Backscatter S_λ and attenuation coefficient β_λ are global parameters optimized along with the MLP. With $\boldsymbol{\alpha}$ and $\mathbf{n}$ we can calculate the reflected radiance l_λ from the scene. We apply a sigmoid function on σ to separate water from scene and calculate transmittance T_λ through the scene and water using different coefficients. With S_λ, T_λ, σ and l_λ, our rendering model predicts the pixel values in the image.

model and propose several approximations for underwater robots.

4.3 Methodology

4.3.1 Neural Scene Representation

We employ neural reflectance field [14] to model the underwater scene observed by an underwater robot with onboard lights. The continuous scene is represented as a function of 3D location $\mathbf{x} = (x, y, z)$ in the global coordinate frame. The outputs of the function are the rendering properties $(\sigma, \boldsymbol{\alpha}, \mathbf{n})$, where σ is the volume density, $\boldsymbol{\alpha} = (\alpha_r, \alpha_g, \alpha_b)$ is the albedo and $\mathbf{n} = (n_x, n_y, n_z)$ is the surface normal (see Fig. 4.3).

In practice, we first sample 3D points $\mathbf{x}$ on camera rays in the global coordinate frame. We then use hash encoding γ to map the input $\mathbf{x}$ into a higher-dimensional space [73] before feeding it into a nested MLP:

$$(\sigma, \boldsymbol{\alpha}, \mathbf{n}) = \mathrm{MLP}(\gamma(\mathbf{x})) \tag{4.1}$$

4.3.2 Rendering Equations

The volume rendering equation [30, 80] maps a camera ray $\mathbf{x} = \mathbf{o} - t\boldsymbol{\omega}$ into the radiance L_λ captured at location $\mathbf{o}$ in direction $\boldsymbol{\omega}$:

$$L_\lambda(\mathbf{o}, \boldsymbol{\omega}) = \int_{t=0}^{d} T_\lambda(\mathbf{x})\sigma(\mathbf{x})l_\lambda(\mathbf{x})dt \tag{4.2}$$

Here T_λ is the transmittance from $\mathbf{x}$ to $\mathbf{o}$, σ is the volume density, l_λ is the scattered radiance from $\mathbf{x}$ to $\mathbf{o}$ along the ray, and λ indicates the wavelength. In this study, the wavelength is discretized into RGB space that $\lambda \in \{r, g, b\}$ [7].

For a light beam emitted from $\mathbf{x}$ to $\mathbf{o}$, the fraction of light that reaches the camera is described by the transmittance T_λ:

$$T_\lambda(\mathbf{x}) = \exp(-\int_{s=0}^{t} \sigma_\lambda(\mathbf{o} - s\boldsymbol{\omega})ds) \tag{4.3}$$

Here, σ_λ denotes the attenuation coefficient as a function of the 3D location $\mathbf{o} - s\boldsymbol{\omega}$, which combines the extinction of light due to both volume-density-dependent out-scattering and wavelength-dependent absorption [30, 43]. The formulation of σ_λ will be further discussed in 4.3.3.

The scattered radiance l_λ from the scene, as a part of the integrand in Eq. 4.2, is formulated as follows:

$$l_\lambda(\mathbf{x}) = \int_{S^2} f_\lambda(\mathbf{x}, \boldsymbol{\omega}, \boldsymbol{\omega}_i)I_\lambda(\mathbf{x}, \boldsymbol{\omega}_i)d\boldsymbol{\omega}_i \tag{4.4}$$

where S^2 represents the spherical domain around point $\mathbf{x}$, f_λ is the phase function that governs the distribution of light scattered at $\mathbf{x}$, and I_λ is the incident radiance from direction $\boldsymbol{\omega}_i$ into $\mathbf{x}$.

In practice, we follow the assumptions in [16] that object surfaces underwater are Lambertian, which scatter light in all directions equally. Following Lambert's cosine law, the phase function for objects underwater is described as: $f_\lambda(\mathbf{x}, \boldsymbol{w}, \boldsymbol{w}_i) = \alpha_\lambda(\mathbf{x})\cos(\mathbf{n}(\mathbf{x}), \boldsymbol{\omega}_i)$. Here $\alpha_\lambda(\mathbf{x})$ and $\mathbf{n}(\mathbf{x})$ are the albedo and normal at $\mathbf{x}$ estimated by the neural network. In other words, we are not modeling any specular reflection which is rare underwater.

Inferring the the phase function f_λ of water volumes, i.e. VSF, is challenging and not scalable on real robots due to different light and camera configurations. To address this, we propose approximating backscatter in the image as a constant and moving away from estimating VSF (see 4.3.4), by which the complexity of our approach is significantly reduced while still achieving accurate and realistic rendering results.

Similar to [16], we only consider direct illumination from onboard lights. While natural and

ambient light also impacts the lighting in shallow water, they are out of the scope of this work. The direct illumination on point $\mathbf{x}$ from the light source is expressed by:

$$I_\lambda(\mathbf{x}, \boldsymbol{\omega}_i) = T^i_\lambda(\mathbf{x}) E^i_\lambda(\mathbf{x}) \tag{4.5}$$

Here i indicates the light source from direction $\boldsymbol{\omega}_i$, T^i_λ is the transmittance from the light source to $\mathbf{x}$ (the calculation is similar to Eq. 4.3), and $E^i_\lambda(\mathbf{x})$ is the intensity of light source i evaluated at $\mathbf{x}$ taking light fall-off with distance into account.

4.3.3 Unified Transmittance Model

The attenuation of light in water can be modeled with a transmittance term T_λ given attenuation coefficient σ_λ and distance t:

$$T_\lambda = \exp(-\int_{s=0}^{t} \sigma_\lambda ds) = \exp(-\sigma_\lambda t) \tag{4.6}$$

Given the emitted radiance E, the arrived radiance is $T_\lambda E$. The attenuation coefficient σ_λ for water can be decomposed into the out-scattering coefficient σ and the absorption coefficient β_λ [43]. Notably, the out-scattering coefficient σ is independent of the wavelength of the light [6], and can be represented as the volume density in rendering equations.

In the neural reflectance field, volume density is a function of spatial location $\mathbf{x}$, so we have:

$$\sigma_\lambda(\mathbf{x}) = \sigma(\mathbf{x}) + \beta_\lambda \tag{4.7}$$

where $\sigma(\mathbf{x})$ is predicted by the neural implicit functions and β_λ will be optimized as a global parameter that doesn't change with spatial locations.

On a camera ray, points in the water attenuate light through both absorption and out-scattering, as described by Eq. 4.7. On the contrary, points on objects have no wavelength-dependent absorption effects. So for underwater scenes $\sigma_\lambda(\mathbf{x})$ can be formulated as follows:

$$\sigma_\lambda(\mathbf{x}) = \begin{cases} \sigma(\mathbf{x}) + \beta_\lambda, & \text{if } \mathbf{x} \text{ is in water} \\ \sigma(\mathbf{x}), & \text{if } \mathbf{x} \text{ is on objects} \end{cases} \tag{4.8}$$

When sampling points from non-transparent objects, the volume density $\sigma(\mathbf{x})$ should typically be

large enough that regardless of whether $\mathbf{x}$ is in water or on objects, $\sigma(\mathbf{x}) \approx \sigma(\mathbf{x}) + \beta_\lambda$. However, it is still important to maintain the separate attenuation coefficients in Eq. 4.8 during training until the prediction of $\sigma(\mathbf{x})$ converges.

To apply Eq. 4.8, we need to differentiate water from the rest of the scene. We experimentally observe that the value of $\sigma(\mathbf{x})$ for objects is at least 10 times greater than that in clear water. This observation also aligns with the measurements by Jerlov [44]. Assuming that there are no highly transparent objects in the scene other than water, we define the following logistic regression functions using the sigmoid function:

$$\begin{aligned} m_o(\mathbf{x}) &= \text{sigmoid}(a(\sigma(\mathbf{x}) - b)) \\ m_w(\mathbf{x}) &= 1 - m_o(\mathbf{x}) \end{aligned} \tag{4.9}$$

where m_o and m_w indicate the probabilities of the query point $\mathbf{x}$ being on non-transparent objects and water, respectively. Specifically, a controls the steepness of the sigmoid function, and a higher value of a results in higher confidence in prediction, but it may also increase the risk of vanishing gradient. b determines the density threshold used to distinguish water from objects. With m_o and m_w, we can express $\sigma_\lambda(\mathbf{x})$ in the following form:

$$\begin{aligned} \sigma_\lambda(\mathbf{x}) &= m_w(\mathbf{x})(\sigma(\mathbf{x}) + \beta_\lambda) + m_o(\mathbf{x})\sigma(\mathbf{x}) \\ &= \sigma(\mathbf{x}) + m_w(\mathbf{x})\beta_\lambda \end{aligned} \tag{4.10}$$

In other words, m_o and m_w can be considered as masks on sample points, exposing those in the water and objects to distinct light-transmitting physics.

4.3.4 Approximating Water Effects

The backscatter effects in water can be described using a VSF. However, in learning neural scene representations from real underwater data, we encounter difficulties in modeling VSF. Firstly, backscatter from the closer regions of the field of view has a greater impact on imaging (Fig. 4.4). We need a precise imaging system model to accurately infer the VSF in this area. This requires detailed information about the dimensions and poses of the camera and light source. However, calibrating such a system complicates the deployment of our algorithm on real robots and is hard to scale across different robot platforms. Secondly, estimating the VSF along the ray prevents us from using bounding planes, which could significantly enhance the sampling efficiency and

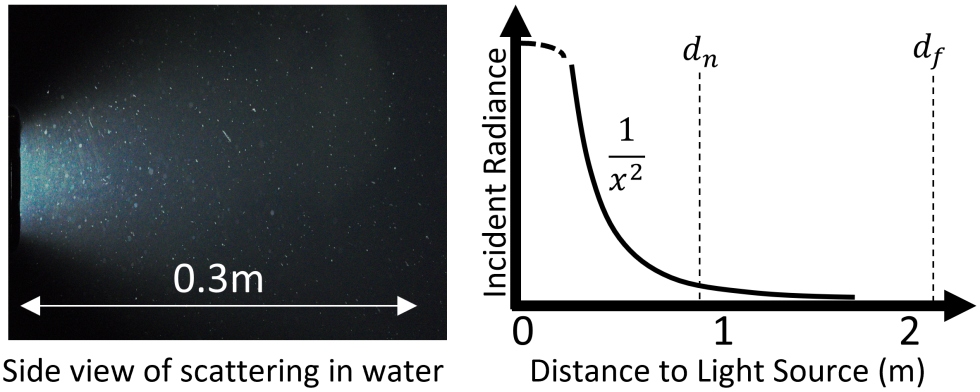


Figure 4.4: A side view of scattering generated from an LED light (left) reflects the intensity distribution of incident radiance. We observe significant light fall-off with the distance from the light source. The plot on the right sketches a typical light fall-off curve. d_n and d_f indicate the typical positions of near and far bounding planes. When the distance is close to the dimensions of the lighting component, we need to precisely calibrate the lighting and imaging components to approximate the curve. The rest of the curve can be approximated with the inverse-square law.

avoid overfitting by constraining the viewing frustum from multiple views. To address the issues mentioned above, we propose several approximations to avoid modeling VSF:

Backscatter as a constant

The backscatter captured in the image can be approximated as a constant S_λ, as the majority of backscatter comes from the region close to the light source, which is not affected when the images are taken from different depths and perspectives (see Fig. 4.4).

Co-centered camera and light source

Points are only sampled between the near and far bounding planes, and their distances to the camera are sufficiently large compared to the typical dimensions of the components of the imaging system. Therefore, we model the light source as a single point light source that is co-centered with the camera, similar to [14]. We use the inverse-square law to calculate the incident radiance $E_\lambda(\mathbf{x})$.

We design a loss function that enforces the model to output $\sigma(\mathbf{x}) = 0$ if $\mathbf{x}$ is in water (see 4.3.6). With this constraint, we can avoid double-count backscatter with both S_λ and Eq. 4.2 since the integrand in Eq. 4.2 will have zero values for $\mathbf{x}$ in the water. Additionally, constraining $\sigma(\mathbf{x}) = 0$

for $\mathbf{x}$ in water allows us to calculate the attenuation between the near bounding plane and the camera without sampling points. As a parameter to be optimized in training, β_λ will approach $\sigma_\lambda(\mathbf{x})$ when $\sigma(\mathbf{x})$ approaches 0 according to Eq. 4.7. Then the transmittance between the near bounding plane and the camera will be $T_\lambda^n = \exp(-\beta_\lambda d_n)$ according to Eq. 4.6, and Eq. 4.2 can be written as:

$$L_\lambda(\mathbf{o}, \boldsymbol{\omega}) = S_\lambda + T_\lambda^n \int_{t=d_n}^{d_f} T_\lambda(\mathbf{x})\sigma(\mathbf{x})l_\lambda(\mathbf{x})dt \tag{4.11}$$

Here d_n and d_f are the distances from the camera to near and far bounding planes respectively.

4.3.5 Ray Marching

We numerically estimate Eq. 4.11 by ray marching. Rays are sampled from the center of the camera and pass through uniformly sampled points on the image plane in training. Points are then sampled along the ray between the near and far bounding planes. The rendering equation is discretized as follows:

$$\begin{aligned} L_\lambda(\mathbf{o}, \boldsymbol{\omega}) &= S_\lambda + T_\lambda^n \sum\nolimits_{i=0}^{N} T_\lambda(x_i)\Phi_\lambda(x_i)l_\lambda(x_i) \\ T_\lambda(x_i) &= \exp(-\sum\nolimits_{j=0}^{i} \sigma_\lambda(x_j)\delta_j) \\ \Phi_\lambda(x_i) &= \frac{\sigma(x_i)}{\sigma_\lambda(x_i)}(1 - \exp(-\sigma_\lambda(x_i)\delta_i)) \\ l_\lambda(x_i) &= T_\lambda^n T_\lambda(x_i) E_\lambda(x_i)\alpha_\lambda \cos(\mathbf{n}(x_i), \boldsymbol{\omega}) \end{aligned} \tag{4.12}$$

where δ_i denotes the step size at sample point x_i. It is worth noticing that transmittance terms T_λ^n and $T_\lambda(x_i)$ are used in both the calculation of incident radiance l_λ and the sensed radiance L_λ according to approximation 4.3.4.

The opacity Φ_λ corresponds to the term $1 - \exp(\sigma(x_i)\delta_i)$ in NeRF and its variants. In NeRF, the volume density σ governs both the emission and attenuation of the radiance, making it sufficient to model objects in the air, haze, and even transparent glowing gas [63]. In our study, we need to model the wavelength-dependent attenuation, which requires both the volume density σ and the attenuation coefficient σ_λ to play a role together in Φ_λ. However, if σ_λ in the denominator approaches 0 in training, the model will encounter numerical issues. To avoid this, we take advantage of our proposition in 4.3.4 that enforces $\sigma(x_i) = 0$ if x_i is in the water, so $\Phi_\lambda(x_i) = 0 = 1 - \exp(-\sigma(x_i)\delta_i)$. When x_i falls on objects, $\sigma_\lambda(x_i) = \sigma(x_i)$ according to Eq. 4.10, so

$\Phi_\lambda(x_i) = 1 - \exp(-\sigma(x_i)\delta_i)$. We then simplify $\Phi_\lambda(x_i)$ into the following form, which is identical to the opacity term in NeRF [66]:

$$\Phi_\lambda(x_i) = 1 - \exp(-\sigma(x_i)\delta_i) \tag{4.13}$$

4.3.6 Loss Function

We use L_2 loss to optimize the rendered radiance with captured pixel values from the raw image, which has linear color. As a result, the L_2 loss will be dominated by errors in the brighter parts of the image, and the darker parts will have a poor rendering quality. To achieve better visual results, we apply a stronger penalization on errors in the darker parts of the image by tone-mapping ψ on both the model output and raw pixel values before passing them into the loss function $\mathcal{L}$ as suggested by [68]:

$$\mathcal{L} = \sum_\lambda \sum_{r \in R} \|\psi(\hat{L}_\lambda(r)) - \psi(L_\lambda(r))\|_2^2 \tag{4.14}$$

Here R is the sampled ray batch, $\hat{L}$ is the raw pixel value and L is the radiance predicted from the model. We use the gamma correction proposed in [2] as our ψ function to map the linear color to sRGB space.

As proposed in 4.3.4, we want to constrain the volume density $\sigma(\mathbf{x}) = 0$ for $\mathbf{x}$ in water. We first set $\sigma(\mathbf{x}) = 0$ for $\mathbf{x}$ in water by multiplying $m_o(\mathbf{x})$. This gives us the refined volume density $\bar{\sigma}(\mathbf{x})$:

$$\bar{\sigma}(\mathbf{x}) = m_o(\mathbf{x})\sigma(\mathbf{x}) \tag{4.15}$$

Then we are able to calculate the refined radiance $\bar{L}_\lambda(r)$ with equations in 4.3.5 using $\bar{\sigma}(\mathbf{x})$ in the place of $\sigma(\mathbf{x})$. The refined loss $\bar{\mathcal{L}}$ is calculated similarly to Eq. 4.14:

$$\bar{\mathcal{L}} = \sum_\lambda \sum_{r \in R} \|\psi(\hat{L}_\lambda(r)) - \psi(\bar{L}_\lambda(r))\|_2^2 \tag{4.16}$$

The total loss is $\mathcal{L}_{total} = \mathcal{L} + \bar{\mathcal{L}}$. By optimizing $\mathcal{L}_{total}$, we are encouraging the model to generate the same results with $\sigma(\mathbf{x})$ and $\bar{\sigma}(\mathbf{x})$. So the prediction of $\sigma(\mathbf{x})$ from network will converge to $\bar{\sigma}(\mathbf{x})$, where for $\mathbf{x}$ in the water, $\sigma(\mathbf{x}) = 0$.

4.3.7 Re-rendering with True Color

To re-render the image with true color, we just need to remove the backscatter S_λ, wavelength-dependent absorption β_λ and volume density $\sigma(x)$ for $\mathbf{x}$ in water. We only need to use $\bar{\sigma}(\mathbf{x})$ in calculating transmittance T and opacity Φ. The rendering equation in 4.3.5 becomes the following:

$$\begin{aligned} L_\lambda(\mathbf{o}, \boldsymbol{\omega}) &= \sum_{i=0}^{N} T(x_i)\Phi(x_i)l_\lambda(x_i) \\ T(x_i) &= \exp(-\sum_{j=0}^{i} \bar{\sigma}(x_j)\delta_j) \\ \Phi(x_i) &= 1 - \exp(\bar{\sigma}(x_i)\delta_i) \\ l_\lambda(x_i) &= T(x_i)E_\lambda(x_i)\alpha_\lambda \cos(\mathbf{n}(x_i), \boldsymbol{\omega}) \end{aligned} \tag{4.17}$$

4.4 Experiments

4.4.1 Data Collection

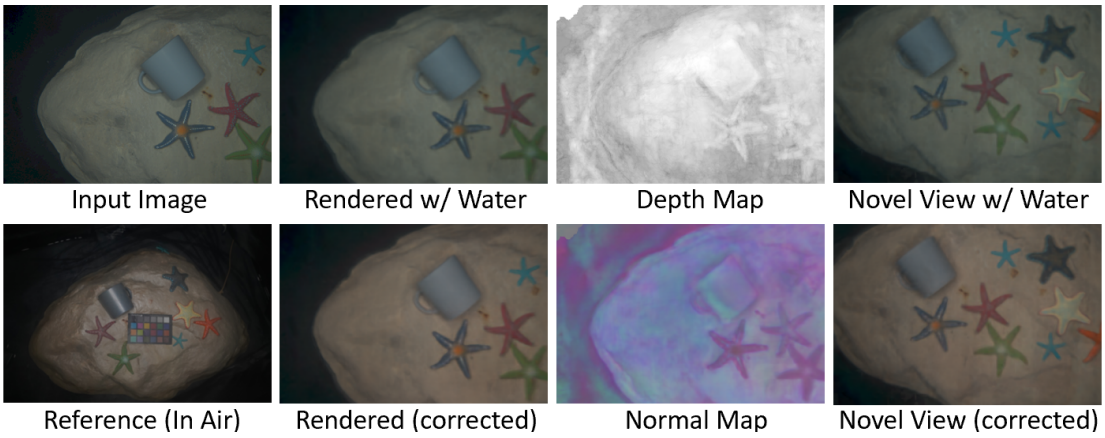


Figure 4.5: An example of our color correction algorithm being applied on the data collected in a water tank. Visualization show that our proposed method is able to recover the color and geometry of the underwater scene, and render image with consistant appearance from novel views.

We collect our underwater data in a water tank with 1.3m water depth (An example is shown in Fig. 4.5), as well as in natural water (Lake Erie) with our underwater robot platform. Our imaging system consists of a Sony ILCE-7M3 camera with a 40mm prime lens and LED lights. The imaging system is housed in a waterproof case and fully submerged when collecting data. The images are

captured using 1/250s exposure time, $f/5.6$ aperture, and ISO 1600. The raw image files with 14-bit pixel values in HDR space are decoded, denoised, and scaled into 8-bit images with linear values using RawPy [92]. We placed artificial decorations with various colors on the bottom of the water tank together with a Macbeth ColorChecker [64]. We use the manufacturer's (X-rite) software to balance the image color as ground truth in quantitative study, which is only used for comparison purposes and does not play a role in our proposed algorithm. We acquire camera poses from COLMAP [100] with post-processed JPEG images to ensure high feature quality. We also build our synthetic data based on implementations from [72, 106] and measurements from [44, 79].

4.4.2 Implementations

Our code is developed using PyTorch3D Library [89]. We use hash encoding proposed in iNGP [73] for positional encoding. We choose $a = 3$ and $b = 3$ empirically for our sigmoid function in Eq. 4.9 (same a, b values are used for all experiments). Our neural implicit function consists 3 sub-MLP predicting σ, $\boldsymbol{\alpha}$, and $\mathbf{n}$ respectively similar to S^3-NeRF [123]. We use `LeakyReLU` between consecutive linear layers and `SoftPlus` as the final layer in predicting σ and $\boldsymbol{\alpha}$ to guarantee non-negative outputs.

The model is trained on an Nvidia RTX 4090 GPU with 24GB memory. In each training iteration, we sample 1000 rays from one image and 100 points on each ray. The model is trained for $50k$ epochs for each scene.

4.4.3 Comparisons

We compare our results with grayworld algorithm [17], histogram equalization [81], FUnIE-GAN [42] and WaterNeRF [101] (we use open-sourced Sinkhorn loss implementation in GeomLoss Library [28]). Results are shown in Fig. 4.6 and Fig. 4.7. Grayworld algorithm and histogram equalization algorithm only correct color well on Synthetic 1 sequence, in which the object's albedo is dominated by low-saturation colors. Under such circumstances, the grayworld and histogram-equalizing assumptions align well with the true color distribution of the scene, so they perform well. However, when we change the body color of the bulldozer to bright yellow (Synthetic 2), grayworld algorithm and histogram equalization are getting downgraded as their assumptions fail. We can observe the same in real images where the albedo of the scene is dominated by a sand-colored background. Grayworld and histogram equalization algorithms both tend to balance

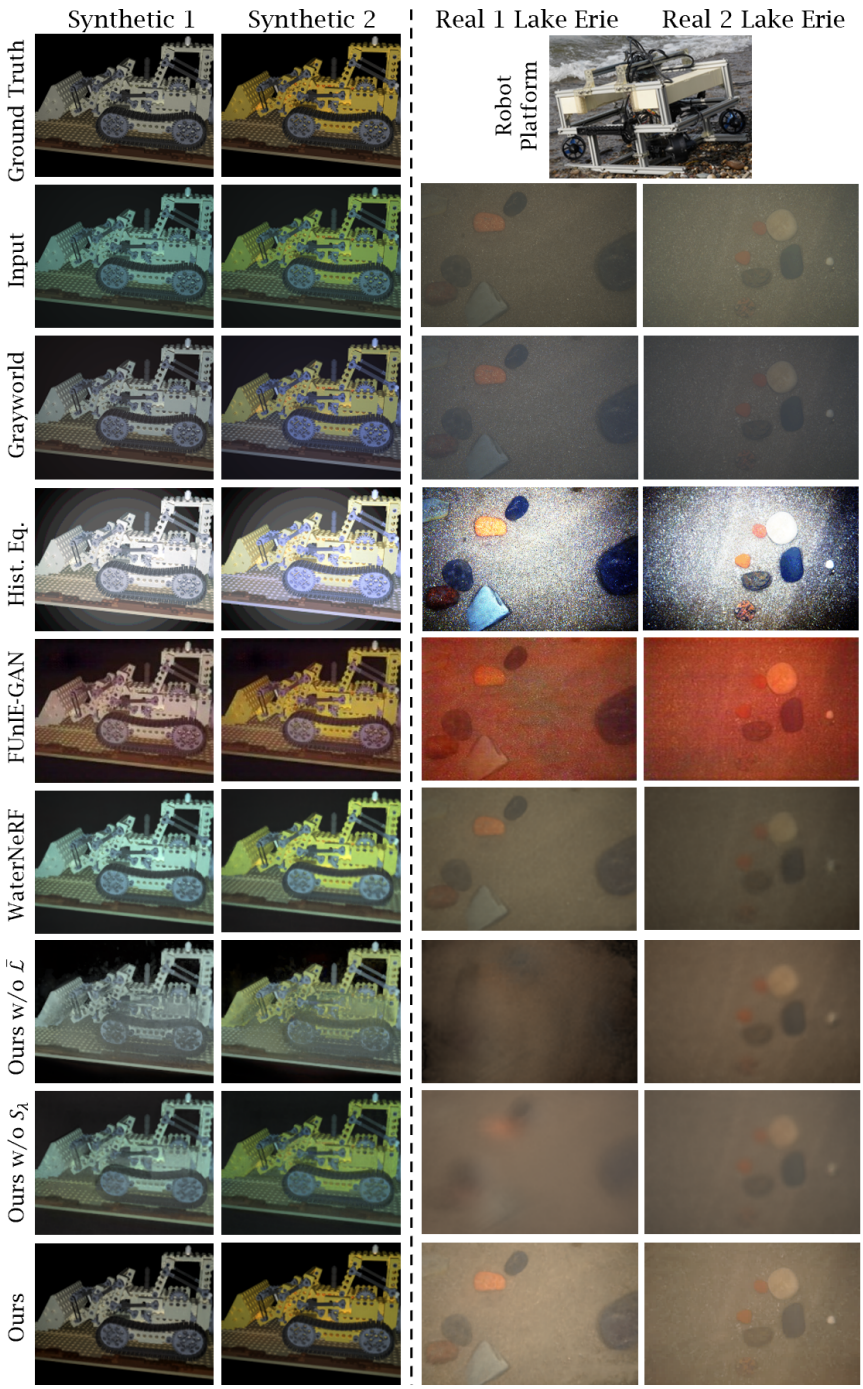


Figure 4.6: Visualizations of color restoration. For good visualization quality, real images are visualized in sRGB space. Reference images for synthetic data are generated by rendering without any water effects.

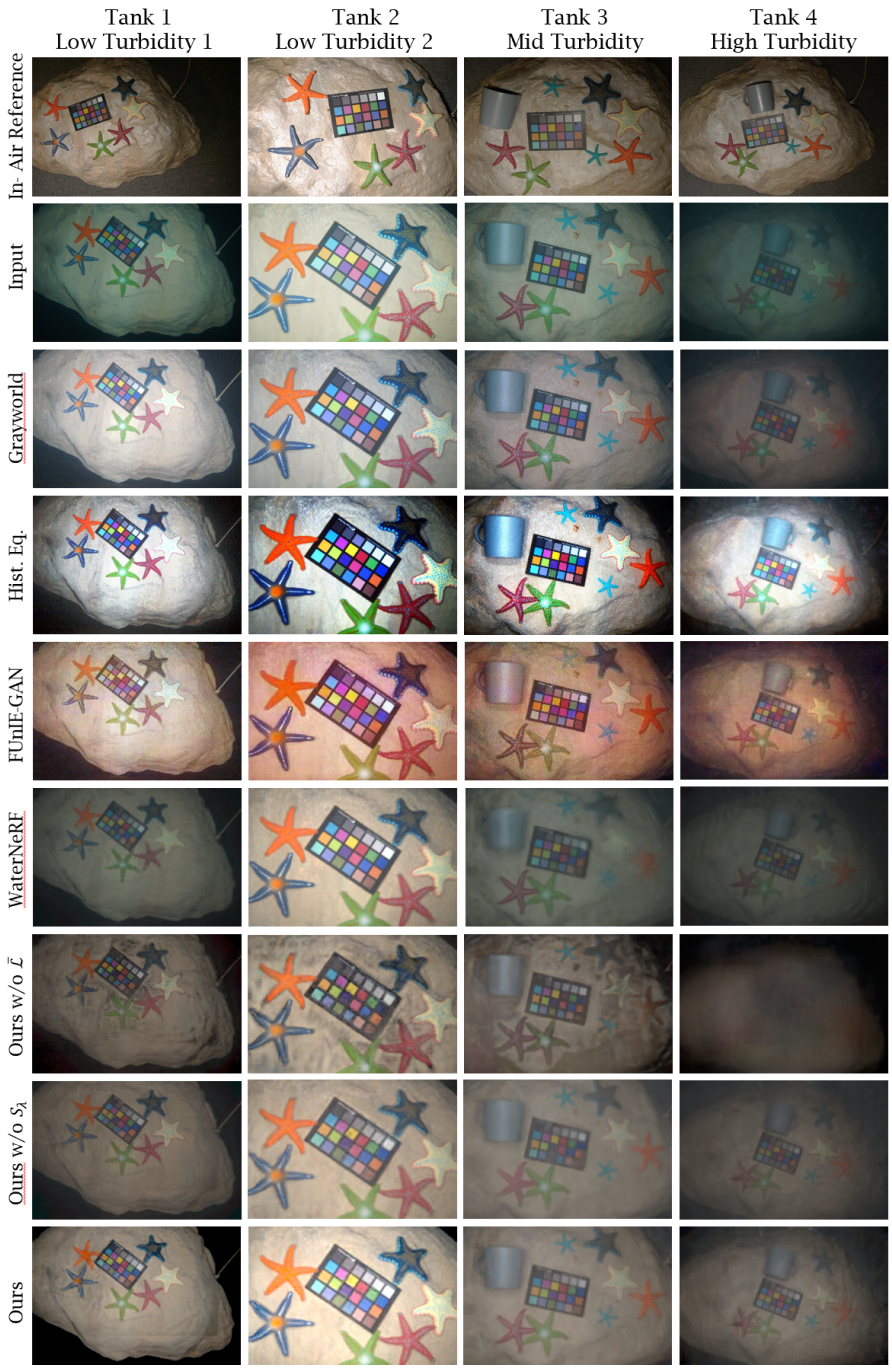


Figure 4.7: Visualizations of color restoration at different turbidity. Images are collected in a water tank with reference image corrected with color chart.

it into grey color. We also observe that both predictions from grayworld and histogram equalization algorithm unpredictably add more veiling light effects or noises into the raw image.

Table 4.1: MSE of A/B channels in CIELAB Space ↓ (pixel values range 0-255)

	Syn. 1	Syn. 2	Tank 1	Tank 2	Tank 3	Tank 4
	A / B	A / B	A / B	A / B	A / B	A / B
Grayworld	4.66/19.7	22.1/91.4	10.7/82.3	**15.1**/110	18.5/104	5.59/9.06
Hist. Eq.	5.94/23.7	21.5/63.5	15.7/87.0	43.6/110	23.6/102	18.8/31.7
FUnIE-GAN	75.5/30.0	61.2/36.1	33.8/61.5	97.4/49.5	95.2/**40.3**	103/61.7
WaterNeRF	55.7/10.3	60.2/13.7	20.9/77.6	15.4/31.1	39.5/96.3	10.1/3.73
Ours w/o $\bar{\mathcal{L}}$	22.8/18.8	60.8/41.1	14.1/74.0	21.1/47.7	21.0/91.4	4.55/4.53
Ours w/o S_λ	45.3/8.73	71.8/30.3	11.6/70.0	20.6/41.3	19.3/92.4	**2.82/3.06**
Ours	**1.15/2.39**	**4.08/9.01**	**9.68/42.5**	19.2/**30.4**	**18.2**/89.9	3.21/3.96

Table 4.2: Angular Error in sRGB Space ↓ (radians)

	Syn. 1	Syn. 2	Tank 1	Tank 2	Tank 3	Tank 4
Grayworld	0.0724	0.2186	0.1381	0.0962	0.1299	0.0563
Hist. Eq.	0.0758	0.2482	0.1421	0.1916	0.1569	0.0810
FUnIE-GAN	0.1107	0.1166	0.1221	0.1655	0.1607	0.1680
WaterNeRF	0.1403	0.1748	0.1303	0.0596	0.1537	0.0514
Ours w/o $\bar{\mathcal{L}}$	0.1048	0.2215	0.0938	0.0759	0.1246	0.0803
Ours w/o S_λ	0.1121	0.2099	0.0909	0.0728	0.1243	**0.0342**
Ours	**0.0361**	**0.0458**	**0.0837**	**0.0591**	**0.1119**	0.0404

As one of the latest GAN-based methods, FUnIE-GAN is pre-trained on annotated underwater images. In our experiments, we find FUnIE-GAN overshooting in the red channel as shown in Fig. 4.6 and Fig. 4.7, implying that color distributions in their training data are less red than ours. In other words, instead of naive assumptions such as histogram equalization, GAN-based methods learn a color distribution from pre-collected datasets. The inherent color distribution in the data for pretraining can deviate from observations as well.

WaterNeRF tackles the problem by applying the physical constraints from Jaffe-McGlamery model while approaching the histogram-equalized image. We acknowledge that it's not a fair comparison since WaterNeRF works for any kind of illumination while our data is only for situations where light and camera move as a rigid body. We observe that when the histogram-equalized image is flawed, e.g. with our synthetic data, WaterNeRF can be significantly downgraded. We also find our method outperforms WaterNeRF in color consistency on real data. For example, comparing

Tank 1 and 2 images in Fig. 4.7, which is from the same image sequence, our method restores the color of the rock with better consistency since we model the albedo and light reflection of the scene, while WaterNeRF only models constant radiance, which fails when the light source moves. In general, from the comparisons, our method restores color in both synthetic and real-world data with the most consistent performance.

We present two metrics for quantitative evaluation: MSE of A/B channels in CIELAB space (Table 4.1) and mean angular error [101] in the sRGB space (Table 4.2). CIELAB is designed to approximate human vision in a uniform space [1] and sRGB is the standard colorspace in which the image is presented. For the synthetic dataset, we use the ground truth from the renderer and calculate both metrics directly. For water tank data, we use color checker software corrected images as ground truth.

As revealed by MSE (Table 4.1), our method performs the best on synthetic data on both A/B channels. Among the 4 water tank images evaluated, our method overall performs well on scenes with low to medium turbidity (Tank 1-3). With high water turbidity (Tank 4), our method is downgraded as our approximation of backscatter is not sufficient to handle such water condition. Besides evaluating LAB channels separately, angular error (Table 4.2) reflects the color similarity in the entire RGB space. Results show that our method performs best across all data except the high-turbidity data, which is consistent with the visualizations in Fig. 4.6 and Fig. 4.7.

We did ablation study on our proposed refined loss $\bar{\mathcal{L}}$ and backscatter term S_λ. Results are included in Fig. 4.6 and Fig. 4.7, Table 4.1 and Table 4.2. We found that without $\bar{\mathcal{L}}$, the quality of image reconstruction is severely reduced when water effects are removed. This implies that light scattered from the scene and from water is mixed up in the model learned, so when water effects are being removed from the reconstruction, the image appears to be incomplete. We also observe that without S_λ, quantitative results show downgraded model performance in most testing cases. The only exception is the high-turbidity case (Real 4) in which our approximation of backscatter is not sufficient to properly model heavy scattering effects, thus underperforming the model without S_λ.

4.5 Extension: Novel Water Effect Synthesis

Our proposed model allows us to adding back a different water effects by altering the learning water effect parameters. As shown in Fig 4.8, we first remove the water effects and then set the attenuation and backscatter parameters with a randomly chosen number. Then we get images

synthesized from novel views with novel water effects. This can be applied to underwater robotic simulation and test the robustness of different computer vision related algorithms.

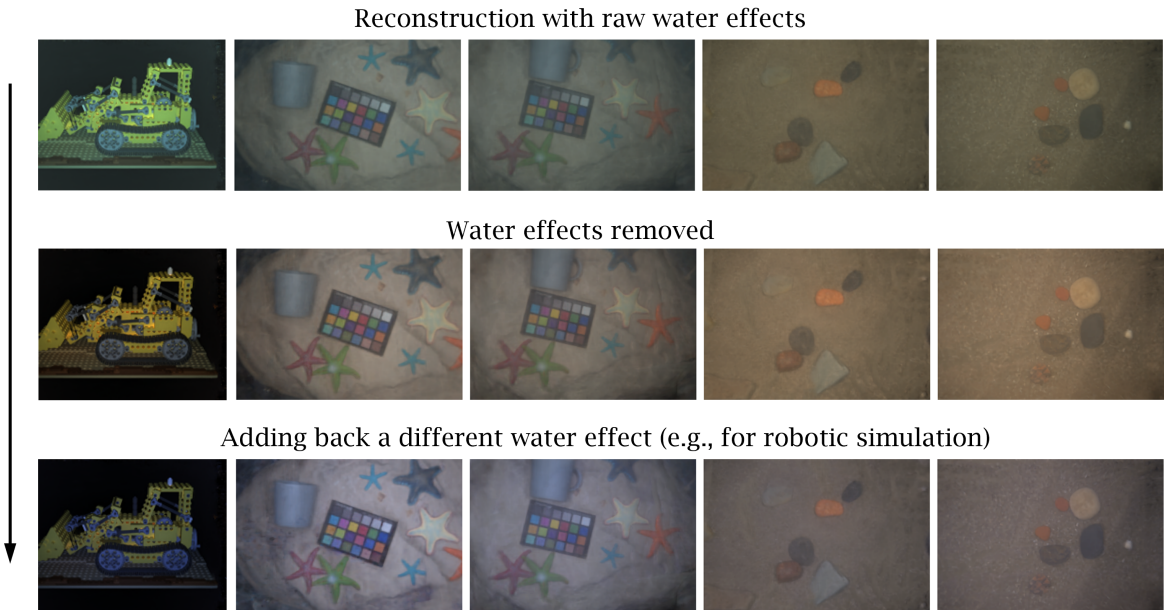


Figure 4.8: Our model allows us to alter the water effects and synthesis images with novel view water effects. Such method can be potentially used in building robotic simulation pipeline with augmented water effects.

4.6 Limitations

This work is directly applicable to underwater imagery collected when dominant light sources move with the camera as a rigid body, such as in deep water, ice-covered water, or cave water. However, it may fail in the following scenarios:

- When the light source is a combination of onboard strobes (point light sources), natural light and ambient light, our model is inadequate for accurately representing water effects from mixed light sources.
- In the presence of highly turbid and layered water, scattering effects vary more significantly with depth, and the robot will have to observe the scene at a closer range (breaking first approximation in 4.3.4). Modeling backscatter as a constant could potentially lead to failure.
- When the baseline between the camera and onboard light source is long, creating shadows

in the observed scene, our model, which assumes co-centered light and camera, cannot accurately represent shadows (breaking second approximation in 4.3.4). This issue also arises with robots equipped with multiple cameras or light sources.

4.7 Conclusion

This work proposes a unified framework that learns underwater neural scene representations together with water effects for underwater robotic imagery. We demonstrate that our method is capable of restoring the true color of the underwater scene with a sequence of observations from different ranges and perspectives. By approximating the backscatter and simplifying the ray tracing, we avoid estimating VSF, which is numerically unstable and requires precise calibration of lighting and imaging system. Additionally, our proposed method generates dense results with end-to-end differentiability and does not rely on any pre-training, depth estimation, or assumptions on prior color distributions.

Future work will extend our model to address the issues discussed in 4.6. Our long-term goal is to achieve true color correction for all types of underwater lighting conditions.

Chapter 5

Camera-Light Source Calibration for Robotic Platforms

5.1 Problem Setup

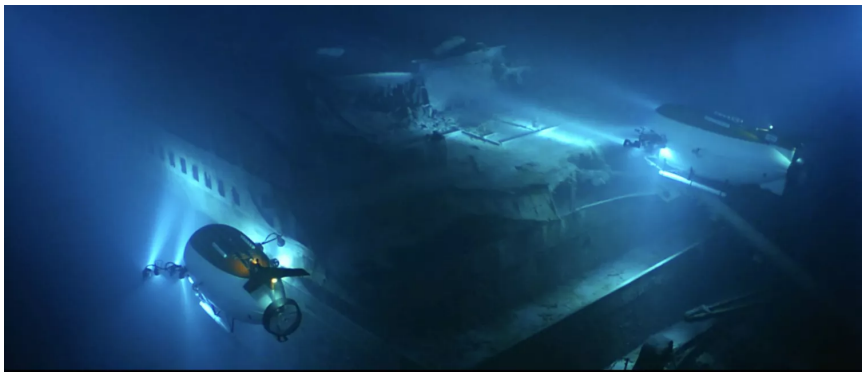


Figure 5.1: A team of underwater vehicles filming the wreck of RMS Titanic in the deep sea, illuminating the scene with onborad light sources.

In the last chapter we developed a system that recovers geometry and appearance of deep water scene with NeRF method, which is based on assumptions including that light source is Lambertian and co-centered with the camera. However, in real-world applications, such assumption rarely holds. An example is shown in Fig. 5.1, that a team of underwater vehicles are exploring the wreck of RMS Titanic. The light sources work in concert with camera systems are: (i) Non-Lambertian,

that the light beam is usually brighter near the centerline; (ii) Non-point light source, that at a close range, the light fall-off does not follow inverse square law (iii) not co-centered with the camera, that a significant baseline can be observed between the camera and the light source.

Building a relightable 3D representation that is similar to the one in Chapter 4 requires ray-tracing among light sources, scene, and camera. So, it is important to know the pose of the light in the camera-light system (extrinsic parameters), and how the light intensity is distributed in the coordinate frame of light source (intrinsic parameters).

In this chapter, we contribute a method called Neural Light Simulator (NeLiS) that learns the camera-light model in a way analogous to camera calibration. We collect calibration data at a customized fiducial target [75] in a dark room. Our proposed algorithm estimates light extrinsics and intrinsics by minimizing the photometric loss between the observed and rendered image. Although this optimization problem is not quite stable in convergence according to our observation, we integrate a GUI into our software that allows human-in-the loop operation, e.g. freezing part of the parameters during training. This effort helps to learn the right model parameters which will be employed in the next chapter for building and relighting 3DGS models in the dark.

5.2 Related Works

Exteroceptive Sensor Calibration on Robots

Autonomous robots are usually equipped with perceptual sensors such as cameras and LiDARs. Taking camera calibration as an example, a calibration target with AprilTag [75] or checkerboard pattern is often used [131]. With observations of the target from different perspectives, the focal length, center of projection, distortion coefficient, and pose of the camera can be estimated [90]. Similar target-based approaches have been used to calibrate LiDARs [13], radars [25] and acoustic sensors [121] for downstream sensor fusion tasks. Analogously to the sensor calibrations mentioned above, in this work, we propose to use a target consisting of AprilTags [75] and blank space to calibrate the light in a camera-light system, including estimating the transformation between camera and light's coordinate system, radiant intensity distribution (RID), and light fall-off curve of the light source.

Light Source Calibration

Light source calibration, especially for understanding the pose of the light source from image observations, has been of interest to the computer graphics community. Existing methods use various kinds of customized calibration targets: [96][97] propose to use a target of AprilTags and pins to infer the position of point light source from the shadows of the pins. Alternatively, [59] [60] propose to use a Lambertian sphere and estimate the light source parameters by learning to reconstruct the sphere. Furthermore, [46] shows that the location of the light source can also be recovered from the shadow of a sphere. With a Lambertian plane and customized markers, [58] shows that it is possible to calibrate a point light source with a single image. [105] shows the most relevance to our work, which calibrates the pose of a light source and the metric scale given the RID curve. However, the above work either assumes an oversimplified model, e.g. a point light source with inverse square light falloff, in a pure dark environment, or that the RID is given, which makes the method less applicable on real-world sensor suites. In this work, we use a minimum calibration target design that consists of only AprilTags [75] and a Lambertian plane similar to [105]. With NeLiS we show that modeling and learning RID, light falloff and ambient light, together with the pose between camera and light source, can improve the quality of image reconstruction and make our calibration method more accurate for photorealistic reconstruction.

5.3 Neural Light Simulator (NeLiS)

The calibration data is taken by capturing photos at a calibration target from different views while the light source moves with the camera as a rigid body. The calibration target is a white plane with four AprilTags positioned as four corners of a rectangle (as shown in Fig. 5.2). We attach the origin of the world coordinate to the top-left corner of the calibration target.

We assume that the camera is precalibrated with distortions removed. One of the key problems for NeLiS to solve is estimating the relative pose of light to the camera $R_l^c \in SO(3)$ and $t_l^c \in \mathbb{R}^3$.

Given the true size of the calibration target, we can apply PnP algorithm [29] to extract $R_c^w \in SO(3)$ and $t_c^w \in \mathbb{R}^3$ which transform observations from the camera coordinate to the world coordinate. The position of the light source is then given by $\mathbf{o}_l = R_c^w t_l^c + t_c^w \in \mathbb{R}^3$. Since we align the z axis of the light coordinate with the centerline of the light, the direction of the centerline can be denoted by $\boldsymbol{\omega}_l = R_c^w R_l^c [0, 0, 1]^\top \in \mathbb{R}^3$. Both $\mathbf{o}_l$ and $\boldsymbol{\omega}_l$ are in the world coordinate frame.

We only use the area bounded by 4 AprilTags as the region of interest (ROI) to do calibrations.

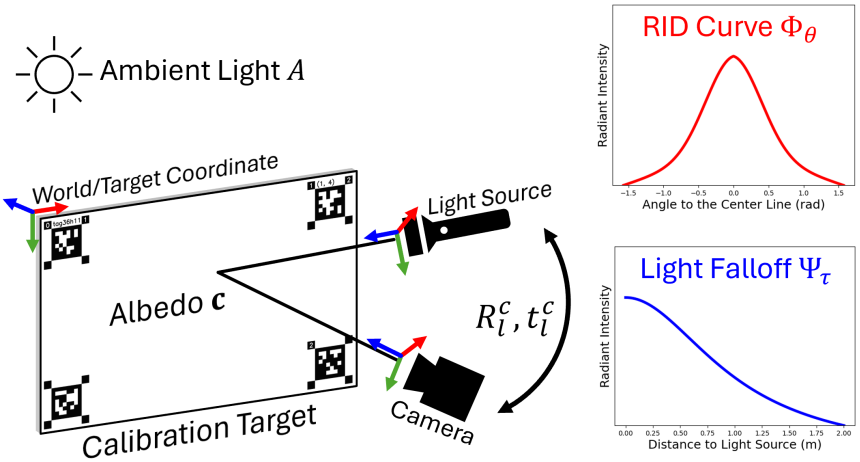


Figure 5.2: NeLiS shading model: Camera poses are localized by AprilTags on the calibration target. The pose of light R_l^c and t_l^c, albedo $\mathbf{c}$ of the calibration target, ambient light A, RID Φ_θ, and light falloff function Ψ_τ will be optimized.

We assume that this area is a Lambertian plane and has the same normal $\mathbf{n} \in \mathbb{R}^3$ and diffusive albedo $\mathbf{c} \in \mathbb{R}^\lambda$ any point on the plane ($\lambda = 1$ for grayscale images and $\lambda = 3$ for RGB). For each pixel in the ROI, we find the intersection of the corresponding camera ray and the target plane in the world coordinate system, denoted by $\mathbf{x} \in \mathbb{R}^3$. To infer the incident radiance at $\mathbf{x}$, one needs to model the RID, light falloff function, and ambient light.

5.3.1 Radiant Intensity Distribution (RID)

RID is commonly modeled as a function of the angle between the centerline $\boldsymbol{\omega}_l$ of the light source and light ray $\boldsymbol{\omega}_\mathbf{x} = \mathbf{x} - \mathbf{o}_l$. Previous work assume that RID is given [58, 60, 105]. However, this assumption may not hold for in-the-wild robot deployment. Instead, we remove this dependency by learning a neural RID Φ_θ from calibration data:

$$\Phi_\theta(\mathbf{x}) = \text{MLP}_\theta(\cos^{-1}(\frac{\boldsymbol{\omega}_\mathbf{x}}{\|\boldsymbol{\omega}_\mathbf{x}\|_2} \cdot \boldsymbol{\omega}_l)) \tag{5.1}$$

here θ denotes the parameters of the MLP.

5.3.2 Light Falloff Curve

Inverse square law is widely used to model light falloff, based on the assumption of a point light source. However, when objects are closer to the light source, the inverse square law starts to fail. We choose to model the light falloff in the form of a Lorentzian function of the distance $\|\boldsymbol{\omega}_{\mathbf{x}}\|_2$, as suggested by [95]:

$$\Psi_\tau(\mathbf{x}) = \frac{1}{\tau + \|\boldsymbol{\omega}_{\mathbf{x}}\|_2^2} \tag{5.2}$$

Instead of estimating τ from hand measurement of the light surface [95], we designate it to be a learnable parameter.

5.3.3 Ambient Light

Imaging and lighting problems are often studied in a dark room to better simplify and constrain the model by removing ambient light. However, a perfectly dark space for calibration might not be accessible for real-world robot deployment. Instead, we model ambient light as a learnable parameter A. The incident radiance at the point $\mathbf{x}$ can thereby be modeled as:

$$I_{\mathbf{x}} = \Psi_\tau(\mathbf{x})\Phi_\theta(\mathbf{x}) + A \tag{5.3}$$

5.3.4 Bidirectional Reflectance Distribution Function (BRDF)

We opt for the Lambertian reflection model and find it sufficient for this work, eliminating the need to optimize any parameters in our BRDF. We use $f_r(\boldsymbol{\omega}_{\mathbf{x}}, \mathbf{n}, \mathbf{c}) = \max(\boldsymbol{\omega}_{\mathbf{x}} \cdot \mathbf{n}, 0)\mathbf{c}$ as our BRDF (Lambert cosine law included in BRDF for simplicity, following convention of NeRV [107]), giving the rendering equation:

$$\hat{L}_{\mathbf{x}} = I_{\mathbf{x}} f_r(\boldsymbol{\omega}_{\mathbf{x}}, \mathbf{n}, \mathbf{c}) \tag{5.4}$$

As mentioned before, $I_{\mathbf{x}}$ is incident radiance at the surface, $\mathbf{x}$ is the light ray direction, $\mathbf{n}$ is surface normal and $\mathbf{c}$ is diffusive albedo of the surface. With captured pixel intensity $L_{\mathbf{x}}$, we use L1 loss and formulate the NeLiS optimization problem as:

$$\min_{\theta, A, \tau, R_l^c, t_l^c, \mathbf{c}} \sum_{\mathbf{x} \in \text{ROI}} \|L_{\mathbf{x}} - \hat{L}_{\mathbf{x}}\|_1 \tag{5.5}$$

5.3.5 NeLiS frontend: Human-in-the-loop calibration

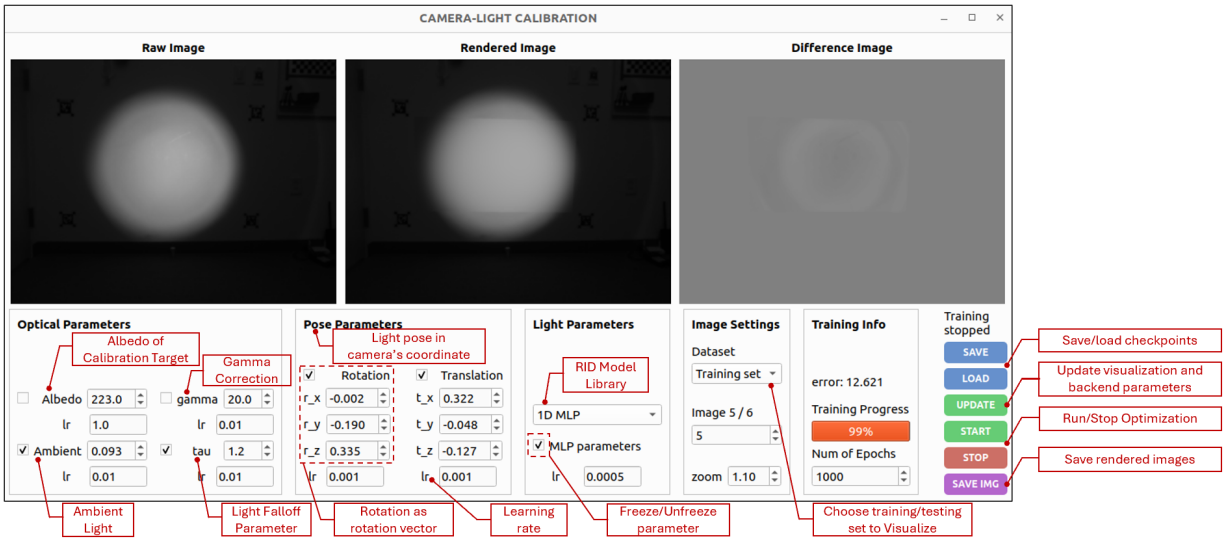


Figure 5.3: GUI of NeLiS, which allows user intervention in the camera-light calibration process.

Although front-end software is not the main contribution of this work, it provides an interactive interface for NeLiS visualization, manual initialization of parameters, and hyperparameter tuning, as partially shown in Fig. 5.3 and elaborated on our github. We find that the NeLiS model is prone to local minimum, especially when learning an MLP and camera-light pose from scratch at the same time. In addition, optimizing MLP and albedo together creates an under-constrained problem, often leading to numerical instability. With our GUI, we first initialize the camera-light pose with a Gaussian distribution as initial RID, and fine-tune the pose, albedo, and other parameters. Then we train an MLP-based RID with pose and albedo frozen, preventing divergence in the concurrent learning of multiple parameters. Once the model has converged, we unfreeze all the learnable parameters and optimize them together to reach the global minimum. Detailed features of our GUI software are illustrated in Fig. 5.3.

5.4 Experiments

5.4.1 Experiments Setup

Our imaging system consists of a FLIR machine vision camera (Model Firefly S) and a light source, together mounted on a rigid structure on top of our legged robot, as shown in Fig. 5.4. The baseline between the camera and the light source is approximately 32 cm, measured by hand. Although the ray-tracing problem can be simplified by placing the light source and camera close enough and modeling them as co-centered [15], in many applications a long baseline between them is a preferred design worth studying. For example, deep-sea robots usually have a long baseline between light source and camera to avoid massive light backscattering from the water.

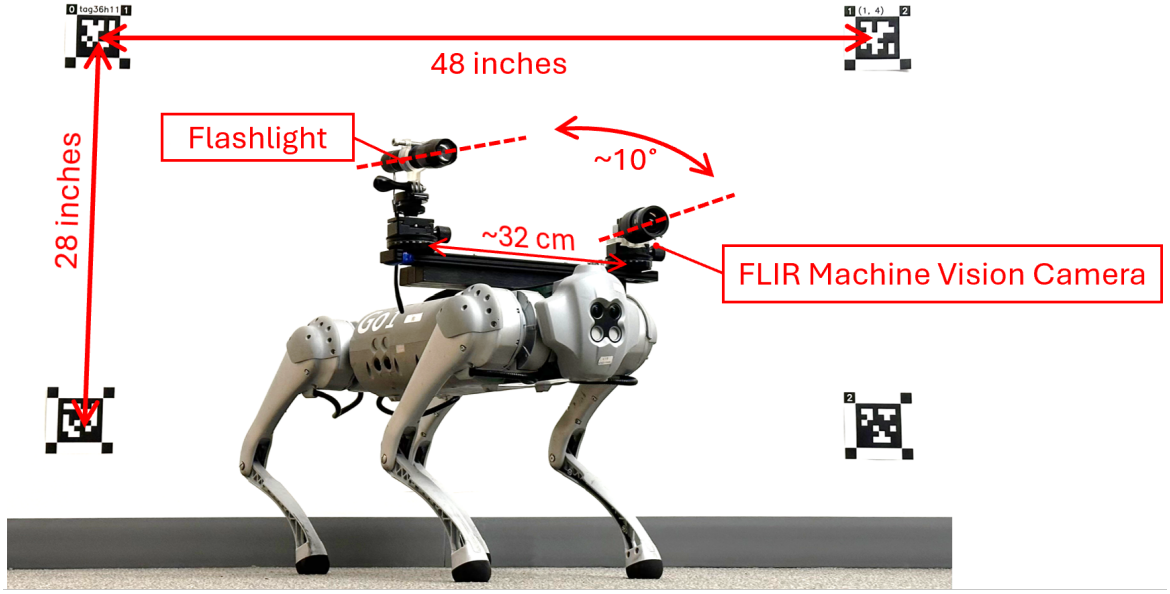


Figure 5.4: Our experiment setup: The imaging system is installed on a legged robot platform (Unitree GO1). We use a FLIR machine vision camera to stream the images in RAW format. The calibration target (as shown behind the robot) is a white wall with four AprilTags positioned in a rectangle.

We experimented with three different light sources: a flashlight, a diving light, and a flood light. To build NeLiS, we take ~ 40 calibration images from different ranges and perspectives for each light configuration and use 25% of the images as a testing set to evaluate the NeLiS performance. Parameters are optimized useing Adam optimizer [50] with LieTorch [110] integration.

5.4.2 Ablation study: RID model

Different light has different RID patterns, which can be observed in Fig. 5.5. Existing methods often model RID as known [105], or general functions such as the power of cosine functions [60] or a Gaussian distribution. However, for most light sources, and all light sources we used in this research, RID is not given. Modeling them with general functions such as a Gaussian distribution will not be adaptive and expressive enough to reflect the true RID of different light patterns for building a photorealistic renderer. The ablation study in Fig. 5.6 (columns 2 and 5) shows that while a Gaussian distribution fits the RID of flood light well, it performs much worse on the other two types of light. In comparison, our MLP-based model fits all 3 kinds of light with the best performance, showing good adaptivity to different light patterns. The result implies that hand-crafted RID models, which are widely used in rendering pipelines, are not sufficient to approximate observations from real-world data lit by a light source with unknown optical behaviors. Using a small neural network can improve the approximation of RID and have the potential to be integrated into photorealistic reconstruction pipelines.

(1) Flashlight **(2)** Flood Light **(3)** Diving Light

Figure 5.5: Light sources on real robots have various RID patterns

5.4.3 Ablation study: light falloff model

For simplicity, previous studies [58, 60, 105] use an inverse square law to model the light falloff, which is based on the assumption that light source is a point light source. However, a real light source has physical size, some of them consists of LED arrays and lens. So a point light source model and inverse square law is not a good fit for real-world light systems. We measure the light falloff of the center point of 3 different light sources as shown in Fig. 5.7. The plot show that when the range is close, light fall-off does not follow the inverse square law as the point light source assumption starts to fail. However, with a Lorentzian model suggested by [95], the light falloff

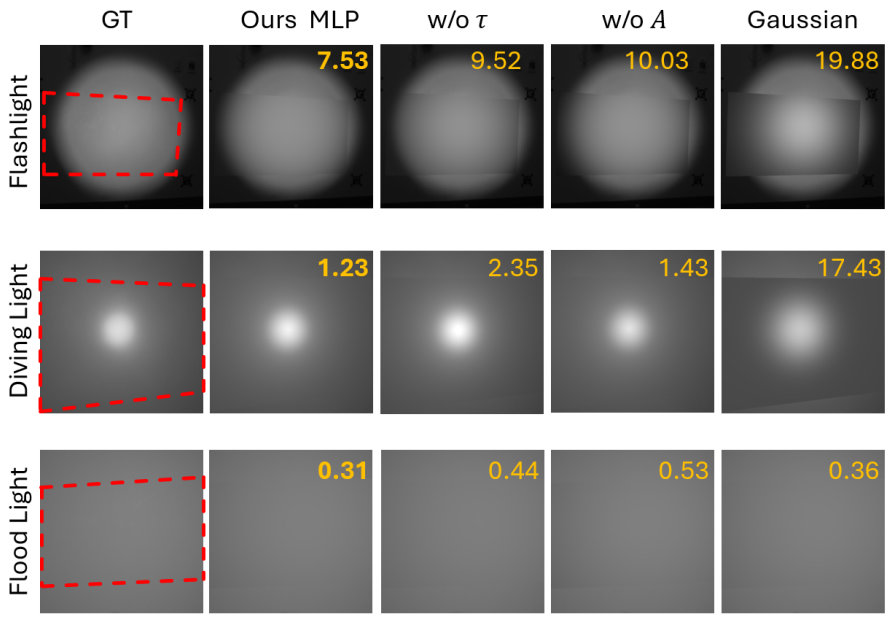


Figure 5.6: Ablation study. The effectiveness of different components in our proposed pipeline is validated. The area bounded in red dashed box is rendered and compared. MSE is highlighted in yellow. Lower MSE means the rendering better approximate captured image.

can be better fitted with our introduced learnable parameter τ. We further show numerical results in Fig. 5.6 (columns 2 and 3) that by learning τ, the rendering performance on testing set of all 3 kinds of lights gets improved, which implies that our light fall-off model better approximates the true light falloff.

5.4.4 Is a perfectly dark environment necessary?

Our approach that estimate the ambient light as a learnable parameter does not require a perfectly dark environment. Ambient light is universal in all kinds of environment and a perfectly dark room for calibration is rare for robot deployments in the wild. Neither our calibration with NeLiS nor deployment experiments with DarkGS (in next chapter) are performed in a perfectly dark environment. As shown in Fig. 5.6 (columns 2 and 4), we show that with our modeling of ambient light A, the performance of the testing set is improved on all three different lights.

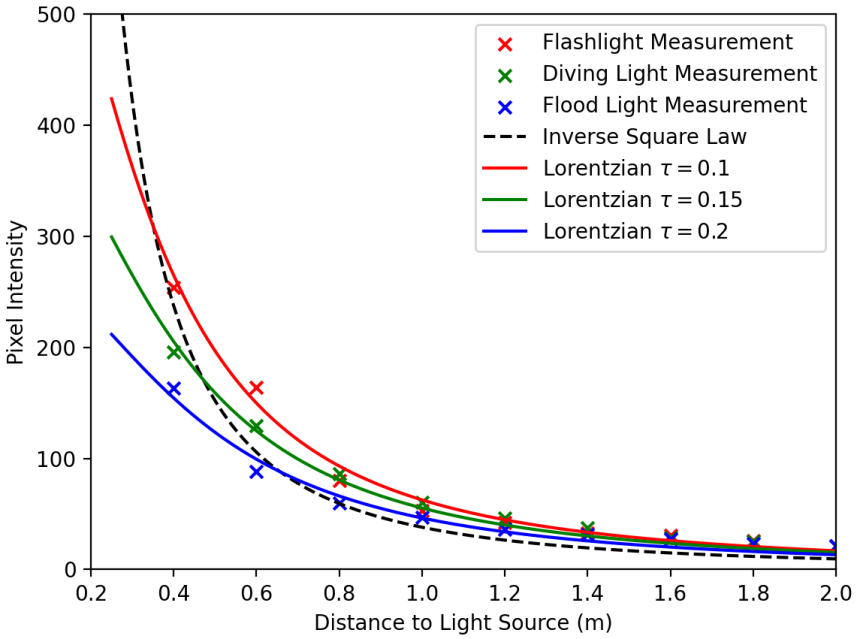


Figure 5.7: Real world measurements of light falloff show that the inverse square law is insufficient to model any of our light sources, but Lorentzian functions [95] with learnable parameter τ fit them well.

5.5 Conclusion

By learning the model parameters of a camera-light system, the proposed NeLiS is able to simulate illumination with photo-realistic quality on the object with known geometry (calibration target). Experiments and ablation show that components in our proposed pipeline effectively improved the rendering quality. Based on NeLiS, we are ready to develop 3D reconstruction methods with such a camera-light system.

Chapter 6

3D Gaussian Splatting for Robots Working in the Dark

6.1 Problem Setup

Figure 6.1: Robotic imaging systems working in the dark consist of cameras and light sources. Examples as shown in (a): Carla Simulator [26], Team CoStar in SubT Challenge [4] and HoloOcean underwater robot simulator [84]. In this work, we propose DarkGS, a 3D Gaussian model that reconstruct the 3D environment from images collected in the dark with onboard light source. DarkGS also allows relighting the 3D model with virtual light source to present the scene under normal lighting effects.

Robots and autonomous vehicles have been routinely deployed in poorly illuminated environments for critical missions and tasks such as exploration, inspection, transportation, search and rescue, etc. (see Fig. 6.1). Imaging systems consisting of one or multiple RGB cameras and light sources are often equipped on the robot to illuminate and sense the surrounding environment. The

Figure 6.2: Our work build 3D Gaussians and relight the scene in dark

streamed image sequence can be further used in downstream tasks, e.g. navigation, mapping, and visualization, to boost the robot autonomy and human understanding of the environment.

Scene reconstruction, or the capability to create accurate internal representations of the environment, is vital for robots operating in unknown environments. Previous vision-only approaches largely rely on identifying common feature points over a set of multiple-view images, and then minimizing a reprojection error [3]. Such procedures like SfM or SLAM also estimate the camera poses of the images. Based on these camera poses, NeRF [66] is capable of achieving photorealistic scene reconstruction by optimizing a photometric loss between the representation and the images. However, while achieving huge success in the graphics community, the transition of objective function from reprojection error to photometric loss has raised a new challenge to the robotics community: Can we still build consistent scene representations with a moving light source on the robot platform?

In this work, we identify *illumination-inconsistency* as the main challenge in building a photorealistic scene representation from images taken with a moving light source. That is, as the robot operates in the environment, the effect of moving light source results in images captured of the same region appearing visually different. Handcrafted feature descriptors that serve as backbone in SfM and SLAM [20, 55, 74] are designed to be invariant to changes in pixel intensities. Such changes also have limited effects on the optimization pipeline, which uses reprojection errors as objective functions. Hence, the classic SfM methods are generally robust against *illumination-inconsistency*. However, on the way towards photorealistic scene reconstruction and switching to optimization of the photometric loss, we find that state-of-the-art NeRF methods [49, 66] fail on the images taken in the dark with a moving light source. Although variants such as RawNeRF [68] and Relightable 3DG [33] claim to be able to handle dark images in RAW format and model the

reflection properties of the scene, they do not take varying illumination into account.

The setup with varying illumination source is also closely related to a well-studied technology called photometric stereo, which refers to the estimation of the shape of an object with multiple images taken from a single viewpoint with different lighting orientations [118]. Early approaches confine this problem to known light source and Lambertian surfaces with uniform albedo. Further work has allowed spatially varying albedo [21] and unknown light [12]. However, our robotic setup differs from this problem by the fact that our camera and light source always move together as a rigid body. And we choose to tackle this problem within the framework of 3DGS [49] which allows us to synthesize novel views with photorealistic quality.

Concretely, the problem tackled in this chapter is as such: Given a sequence of images taken in poorly illuminated environments, with one major light source moving with one camera as a rigid body, reconstruct the scene by minimizing photometric loss and achieve photorealistic novel view image synthesis (Fig. 6.2). We propose DarkGS, a variant of the 3DGS model that builds photorealistic scene representations with onboard light source under poorly illuminated conditions and relights the scene with global illumination, based on NeLiS and COLMAP [100] results.

6.2 DarkGS

6.2.1 Relightable 3D Gaussians

Within the framework of 3DGS [49], we model the scene with a point cloud of Gaussians $\mathbf{G}$ (as shown in Fig. 6.3). Each Gaussian g_i in the point cloud encompasses attributes including position $\mathbf{p}_i$, covariance Σ_i, opacity α_i, albedo $\mathbf{c}_i$ and normal $\mathbf{n}_i$, that $g_i = \{\mathbf{p}_i, \Sigma_i, \alpha_i, \mathbf{c}_i, \mathbf{n}_i\} \in \mathbf{G}$.

Given $\mathbf{p}_i$ and a calibrated light source, the incident radiance I_i can be calculated by Eq. 5.3. With $\mathcal{N}$ ordered points for pixel (u, v), the rendering equation then becomes:

$$\hat{L}_{u,v} = \sum_{i \in \mathcal{N}} I_i f_r(\boldsymbol{\omega}_i, \mathbf{n}_i, \mathbf{c}_i) \alpha_i \prod_{j}^{i-1} (1 - \alpha_j) \tag{6.1}$$

6.2.2 Scale Recovery

The framework of 3DGS and its variants are often based on monocular SfM solutions such as COLMAP [3]. While calibration using AprilTags recovers poses in true metric scale with NeLiS,

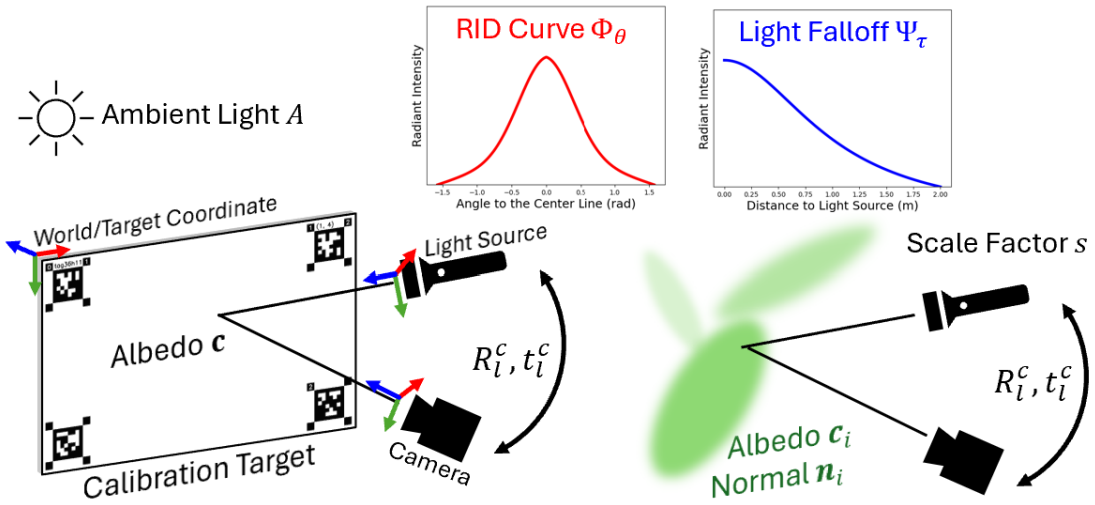


Figure 6.3: Our shading model: (Left) With NeLiS, camera poses are localized by AprilTags on the calibration target. The pose of light R_l^c and t_l^c, albedo $\mathbf{c}$ of the calibration target, ambient light A, RID Φ_θ, and light falloff function Ψ_τ will be learned. (Right) In building DarkGS, each Gaussian g_i is modeled with an albedo $\mathbf{c}_i$ and normal $\mathbf{n}_i$ as learnable parameters. The ambient light A and scale s will also be optimized in this process.

monocular SfM only gives up-to-scale poses for building 3D Gaussians. Here, we introduce a scaling factor $s > 0$ as a learnable parameter so that we can obtain the positions with scale $\mathbf{p}_i' = s\mathbf{p}_i$. With captured pixel intensity $L_{u,v}$, the 3D Gaussian can be built by solving the follow optimization problem:

$$\min_{A,\mathbf{G},s} \sum \|L_{u,v} - \hat{L}_{u,v}\|_1 \tag{6.2}$$

6.2.3 Training Warm-Up

In some cases, a large discrepancy between the initial scale and the true scale often leads to divergence and local minimum at the beginning of optimization. We therefore propose a warm-up strategy to tackle this problem. The overall idea behind this warm-up strategy is that we overwrite our light pose to be co-centered and parallel with the camera. In this co-centered configuration, the baseline between camera and light will be zero, so that the scale difference between the baseline and the point cloud will have a minor effect on shading at coarse level. Then, we gradually recover the pose to the calibration result over the first k iterations. During this process, Eq. 6.2 is being solved and s will be optimized. We define a warm-up factor that grows with iterations, so that in

the m^{th} iteration, this factor is $\frac{m}{k}$. Here, we denote Lie exponential map and Lie logarithm map as follows:

$$\exp(\cdot) : \mathbb{R}^3 \rightarrow SO(3), \;\; \log(\cdot) : SO(3) \rightarrow \mathbb{R}^3 \tag{6.3}$$

so that in warm-up stage we replace R_l^c and t_l^c with:

$$\hat{R}_l^c = \exp(\frac{m}{k}\log(R_l^c)), \;\; \hat{t}_l^c = \frac{m}{k} t_l^c \tag{6.4}$$

In other words, the 3D Gaussian model $\mathbf{G}$ is initialized with $\hat{R}_l^c = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ and $\hat{t}_l^c = [0\ 0\ 0]^\intercal$. Discrepency in the metric scale will have no effects on this setup as the coordinate frames of camera and light source are perfectly aligned. As the iteration grows, the scaling factor s will be optimized, and in the end of warm-up stage we will have $\hat{R}_l^c = R_l^c$ and $\hat{t}_l^c = t_l^c$.

6.2.4 Relighting

Once the DarkGS model is built, we can relight the scene by replacing the components in Eq. 5.3, i.e. Ψ_τ, Φ_θ and A, by carefully designed values and functions. For example, replacing the MLP function in Φ_θ with a constant value can create Lambertian illumination without a strong pattern; replacing the light falloff function Ψ_τ with a constant can simulate the illumination from a parallel light source.

6.3 Experiments

The same camera-light setup in chapter 5 is being used. For building DarkGS, we take $50 \sim 150$ images for each scene. All images are streamed in RAW format without any tone mapping or white balancing. Parameters are optimized using Adam optimizer [50] with LieTorch [110] integration similar to chapter 5.

6.3.1 Failure case of existing 3DGS methods

We first investigated whether existing 3DGS methods can reconstruct the scene using the images we collected with our robotic setup in the dark. We experimented with: 1. vanilla 3DGS [49] which

Figure 6.4: None of the existing methods can solve the problem: Results of Vanilla 3DGS [49], RawNeRF [68] Relightable 3DGS [33] show heavy artifacts and fail to converge. The key reason for the failures is that the existing method does not model the illumination change.

models the scene with constant radiance 2. RawNeRF [68] which is developed to reconstruct scene from RAW HDR images 3. Relightable 3DGS [33] which models the shading and environmental light map. To make a fair comparison, we apply gamma tone mapping to images for vanilla 3DGS [49] and Relightable 3DGS [33] which brightens the image and smooth the discrepancy between illuminated and under-illuminated areas. We also replace RawNeRF's backbone MLP-based renderer [68] with Gaussian render [49].

As shown in Fig. 6.4, we found that none of the existing methods is able to build valid scene reconstructions. We see an excessive amount of artifacts in the center area of the synthesized image. The key reason is that the capability to model varying illumination is missing from existing methods.

6.3.2 Results Visualization

We deployed our system in various real-world environments and the results are shown in Fig. 6.5. Compared with ground truth, our model is able to reconstruct the image with photorealistic quality with learned illumination. Then we replace the light source in our model with a Lambertian light to create a global illumination that illuminates the entire FOV of the camera with photorealistic rendering quality.

Further, we show that our method can be used to synthesize novel-view images, as shown in Fig. 6.6. The images rendered from novel-views have the same visual quality comparing to the training views in Fig. 6.5, and we can do the same relighting to the novel-view images. In addition, we manually adjust the white-balance of the relighted image to approximate human perception.

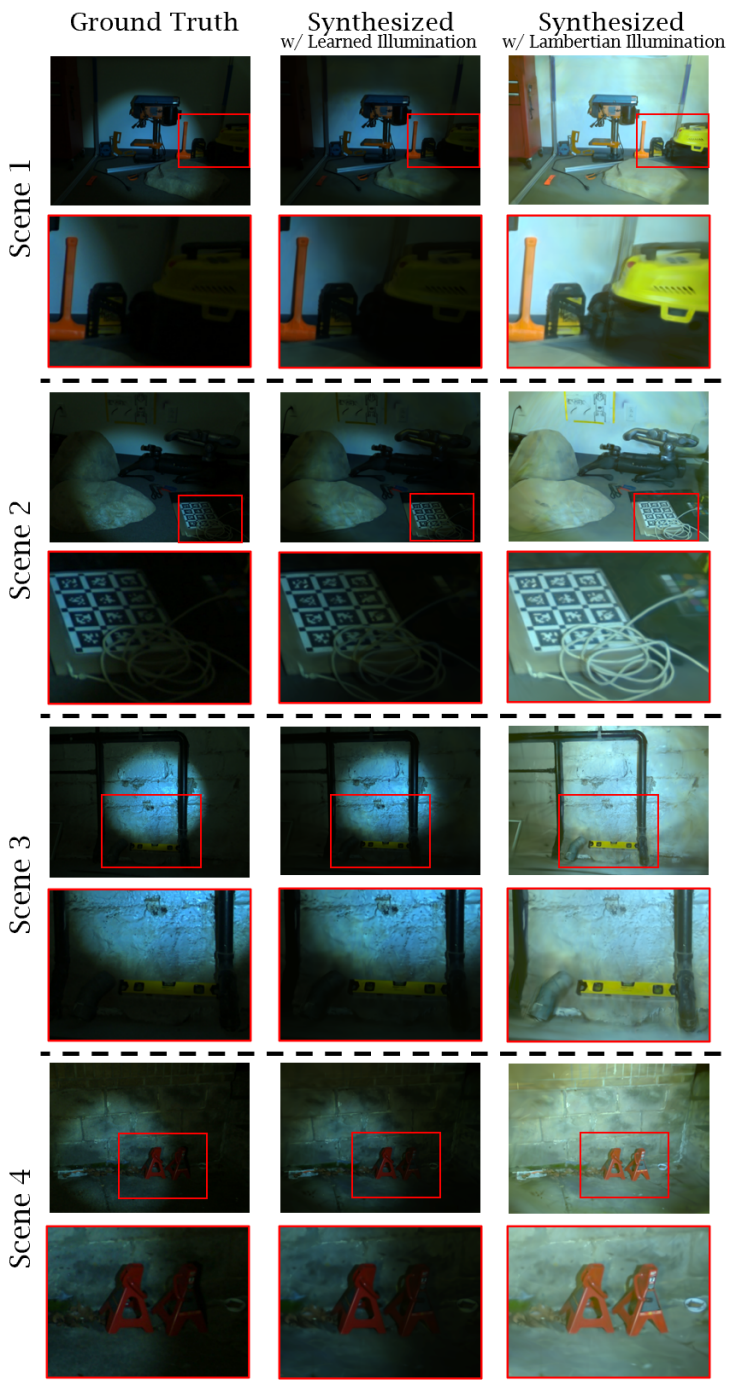


Figure 6.5: Visulization of results from multiple scenes: We show that with DarkGS, we can reconstruct the scene with RAW images from robotic deployments in dark environments, and relight the scene to reveal more information that is corrupted in the RAW image due to uneven and partial illumination. Results as shown are all from the flashlight setup which is the most challenging according to our numerical results.

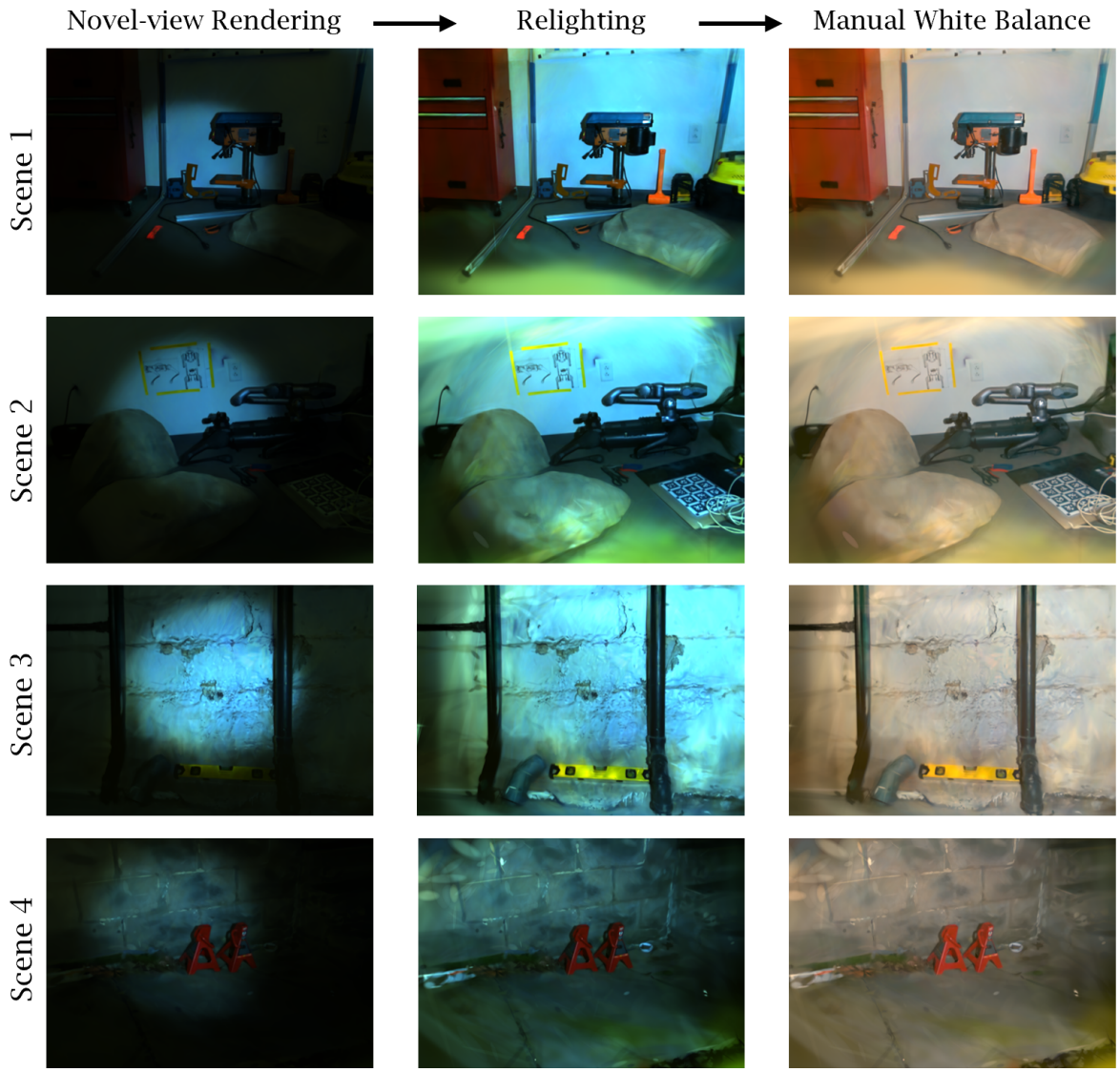


Figure 6.6: Novel-view rendering of scenes with learned light pattern, relighted with a virtual Lambertian light source, and then white-balanced manually.

6.4 Limitation: Shadows

Occlusion is ubiquitous in 3D environments. In dark environments, a moving light source creates dynamic shadows when occlusion is present. However, in our DarkGS model, occlusion is not considered in ray tracing, so shadows observed cannot be modeled. As shown in Figure 6.7, shadows on a flat surface are overfit in our model with 3D structures, to create consistent rendering from multiple views.

Figure 6.7: DarkGS model is not able to model shadows as it does not consider occlusions. Depth map show that dynamic shadows on a flat surface (wall) are overfit with 3D structures.

6.5 Conclusion

This work aims to solve the problem of building 3D Gaussians and scene relighting from images taken by a moving camera-light system. Our proposed pipeline consists of NeLiS, a camera-light simulation and calibration model, and DarkGS which build's photolistic representation for scenes in the dark. The results show that our proposed pipeline can build relightable Gaussians from images taken by the robot platform deployed in the dark, while other current approaches are not able to do the job. Ablations show that components in our proposed model effectively improve the performance of our shading model, so we can learn RID with an MLP for arbitrary light pattern, better approximate the light-falloff curve, and allow us to calibrate and deploy the system in environments that are not perfectly dark. These efforts make our model more applicable in real-world robot deployments. Future work includes modeling shadows and non-Lambertian objects, and investigating the geometry built in the DarkGS model.

Chapter 7

Realistic Underwater Terrain Generation Controlled by Fractal Latents

7.1 Problem Setup

Scene generation is widely studied today, with deep neural networks capable of creating realistic 3D environments trained on large-scale visual data. This technology has a significant impact across various fields, including the film and gaming industries, as well as robotics and autonomous vehicle simulations. In this chapter, we explore the application of deep generative models to the unique setting of underwater environments. Without sufficient data and annotations, the following questions for underwater scene generation remains open:

- What kind of data can we use to train an underwater generative model?
- How can we train the underwater 3D generative model without 3D scans?
- How can we control the sampling process while the data come with no captions or annotations?
- How can we generate underwater terrain with natural-looking variation in appearance?
- What techniques can we use from off-the-shelf 3D generative models and what is lacking in current open-source models?

In this work, we tackle the problem from the perspective of robot perception. Underwater robots and AUVs are designed to travel long distances under the sea, maintaining altitude and

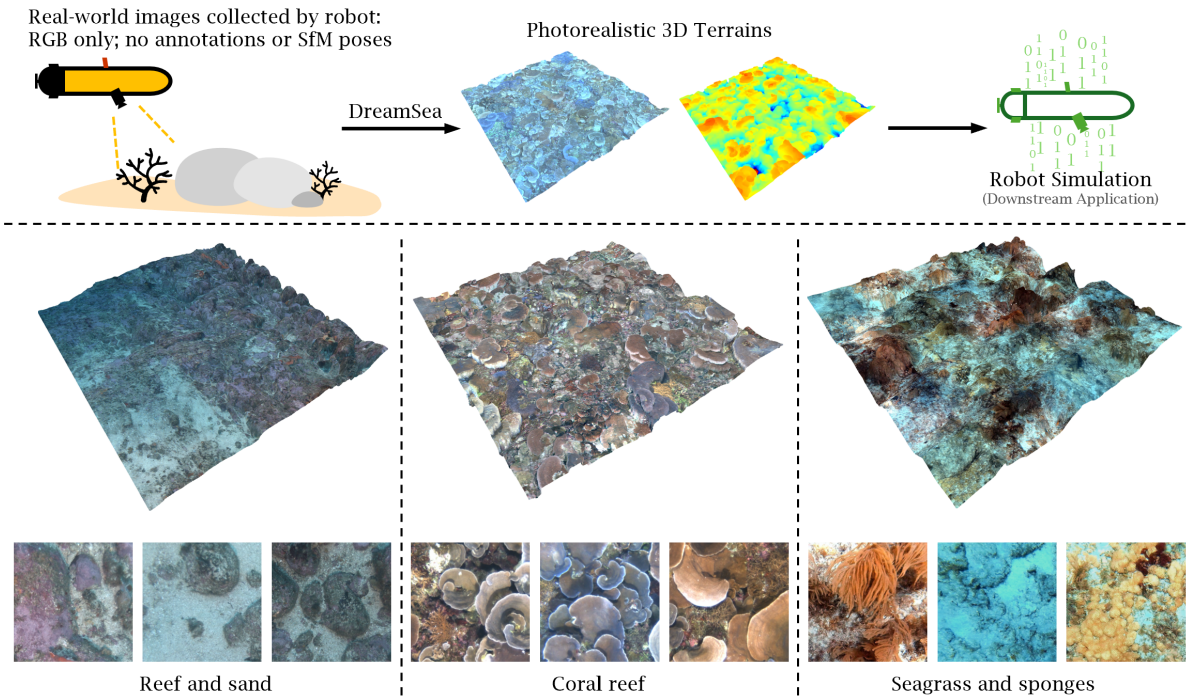


Figure 7.1: **Underwater 3D terrain generation:** Given 2D images of the real world seafloor collected by robots, DreamSea distills 3D geometry and semantic information from visual foundation models and trains a diffusion model that generates realistic 3D underwater scenes conditioned on latent embeddings from a fractal process. All images and maps shown above are synthesized with DreamSea.

route to survey the designated area autonomously. Compared to typical images and videos on the Internet, underwater robotic images cover much larger areas of the terrain. However, the massive amounts of data collected by underwater robots present unique challenges: It is difficult to acquire 3D information directly from sensory streams, as depth sensors and LiDARs commonly do not work well underwater. In addition, natural water bodies are highly dynamic, and visibility is low as a result of light scattering and absorption in the medium. Therefore, SfM [3] and SLAM [74] solutions have unstable performance. As a result, a significant amount of robotic data comes with no camera poses, and the cost of expert annotation is extremely high.

This chapter introduces *DreamSea*, a diffusion-based generative model that can infinitely generate photorealistic 3D underwater scenes. **DreamSea is trained on RGB images captured by underwater robots without any 3D sensory information, SfM poses or human annotations.** After training, scenes generated by DreamSea are spatially consistent in geometry with natural-looking variations in appearance. The contributions of this chapter are as follows:

1. A novel approach that leverages a *fractal* distribution of latent embeddings to control the appearance of generated terrains;
2. Integration of VFMs on unseen underwater images to exploit semantic and 3D geometric information for scene generation; and
3. A pipeline that integrates the state-of-the-art developments image diffusion, inpainting, VFM and 3DGS [49], to allow the generation of photorealistic 3D terrains from unannotated images.

7.2 Related Works

7.2.1 Procedural Terrain Generation

Early studies on procedural terrain generation focus on generating elevation maps that resemble the 3D structure of real-world terrain [69]. In particular, explicit mathematical models such as fBm [61], the diamond square algorithm [32], and Perlin noise [78] are commonly used to approximate natural variations. Modern approaches have enabled the generation of 3D scenes consisting of a variety of assets procedurally and rendered with photorealistic quality [87]. Similar procedural strategies have also been applied to generate room layouts [24] and object-level [37] layouts that can be used to train embodied AI algorithms. However, those modern approaches are based on pre-modeled 3D assets. While it is feasible to specify these assets in advance for commonly seen objects and scenes, e.g. indoor environment, this is not the case for unseen environments such as the deep sea.

7.2.2 Deep Generative Models

Given an image dataset, an image generation model learns the distribution of this dataset. Unseen image samples can be generated as samples drawn from this distribution. Early techniques such as Variational Autoencoders (VAEs) [51] and Generative Adversarial Networks (GANs) [35] are able to generate realistic images. In recent years, models such as DDPM [40], Stable Diffusion [93], TRELLIS [119] and DiT [77] allow high-quality generation that can be conditioned on language inputs. Some of these technologies have also led to commercialized models such as ChatGPT and SORA. We illustrate attempts to generate underwater scenes using large off-the-shelf models in Figure 7.2. Although these models are capable of creating arbitrary scenes, we find, empirically,

that the quality of generated underwater scenes is significantly lower than that of other more common environments. It can be hypothesized that the training data for underwater scenes is scarce and unbalanced. The development of specialized models with curated data for underwater scenes remains an open problem. Our DreamSea model uses a DDPM [40] network with the RePaint [57] framework as a backbone generation and inpainting model.

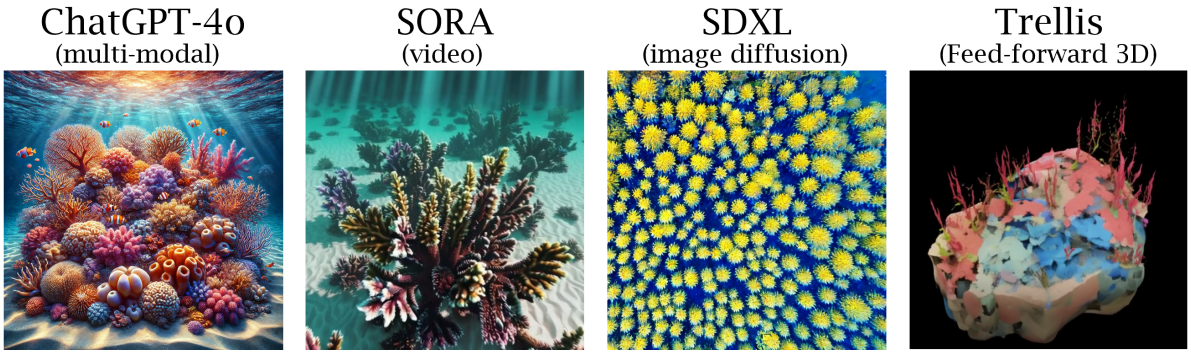


Figure 7.2: Off-the-shelf solution for generating underwater scenes: Generalist generative models [93, 119] are able to generate scenes with diverse appearances, but present heavy artificial effects even though prompted with the “photorealistic style" keyword.

7.2.3 Visual Foundation Models

Underwater robotic field tests typically result in massive amounts of images that are extremely challenging to annotate and often lack 3D information. In this work, we leverage visual foundation models, which are trained on internet-scale data to infer semantic and geometric information by the images collected by our robots. CLIP [86] is a vision-language model (VLM) trained on internet-scale image-caption pairs and generalizes to unseen images. DINOv2 [76] is another foundation model that encodes an RGB image in a vector representation. In this work, we train the image diffusion model conditioned on DINOv2 representations, so the diffusion can be controlled in the latent space. Depth Anything v2 [122] is a depth foundation model that predicts depth from RGB images. In many cases this is used to generate RGB+Depth (RGBD) images from RGB image inputs.

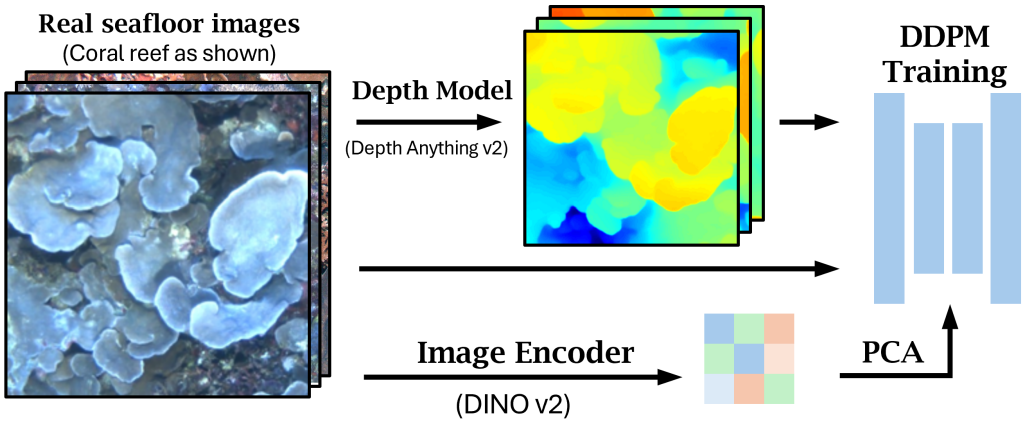


Figure 7.3: **Overview of Training:** Given RGB-only images collected from underwater surveys, we generate depth channels and embeddings with visual foundation models [76, 122]. A DDPM network is then trained with an RGBD image as input conditioned on embeddings.

7.3 DreamSea

At the center of DreamSea is a terrain generation model that varies in spatial coordinates. This model can then generate a set of consistent images spanning a desired spatial region, which can be used to construct 3DGS representations. Particular care needs to be taken to ensure that the generated images reflect both the biological and landscape diversity of marine environments, while being logically consistent across space.

This section elaborates on the design consideration and methodology details of DreamSea, and is structured as follows. 7.3.1 outlines the extraction of relative depth from diverse underwater data from different expeditions. 7.3.2 introduces our diffusion-based generative model that is conditioned on *zero-shot visual features*, enabling the controlled generation on varied underwater environments. 7.3.3 introduces our novel fractal-based generation approach, which enables a set of spatially *consistent* underwater images to be generated and allows explicit control of the diversity of the generated terrain. Finally, 7.3.4 leverages the terrain generated by our generative model to construct a 3DGS representation supervised by the 2D diffusion prior. An overview of our training procedure is outlined in Figure 7.3, and the generation procedure in Figure 7.4.

7.3.1 3D Structure from Depth Foundation Model

To build more consistent 3D structures underwater, we seek to incorporate depth into the diffusion-based generative model. This, however, can be challenging. While traditional 3D reconstruction and mapping methods such as SfM and SLAM have been demonstrated on underwater data, the community struggles to scale up the application of these methods due to challenging underwater environments. These challenges often manifest via low visibility, dynamic surroundings, heavy motion blur under low light, and different sensor set-ups between expeditions to collect data. In this chapter, we use the depth foundation model, Depth Anything v2 [122], to generate a depth map from 2D image data. Depth foundation models are good at predicting the relative depth distribution in single frames. We normalize this prediction to $[0, 1]$. In this work, we consider depths up to a scale factor, and do not require absolute metric depth. The metric scale can be recovered with additional sensors or classic stereo-matching methods. The estimated depths are used as additional depth channels for the real-world RGB training data.

7.3.2 Conditional Diffusion on Zero-shot Features

Underwater robotic images do not come with captions. Additionally, annotating underwater data is also exceedingly challenging and requires a massive expert-level effort. Relying on manual labels would both be costly and difficult to scale. In light of this, we leverage the foundation visual model, DINOv2 [76], to extract zero-shot features from underwater images: for the image data set, we first generate DINOv2 features and then apply Principal Component Analysis (PCA) on the feature set to project high-dimensional features to the low-dimensional space. This reduced dimensional feature vector then acts as a descriptor of the contents within the image. Similar ideas have been explored in LangSplat [85] in which a Variational Autoencoder (VAE) [51] is trained to project CLIP [86] features onto a low-dimension space. Early work by Zhang et al. [126] takes a similar approach on seafloor mapping data with self-supervised training. However, here, by integrating foundation models, we are not required to train large neural networks from scratch to extract features, and can instead apply weights pre-trained on Internet-scale data.

After obtaining a reduced-dimensional feature vector for each image, we train a diffusion model conditional on feature vectors, to generate both RGB and depth images.

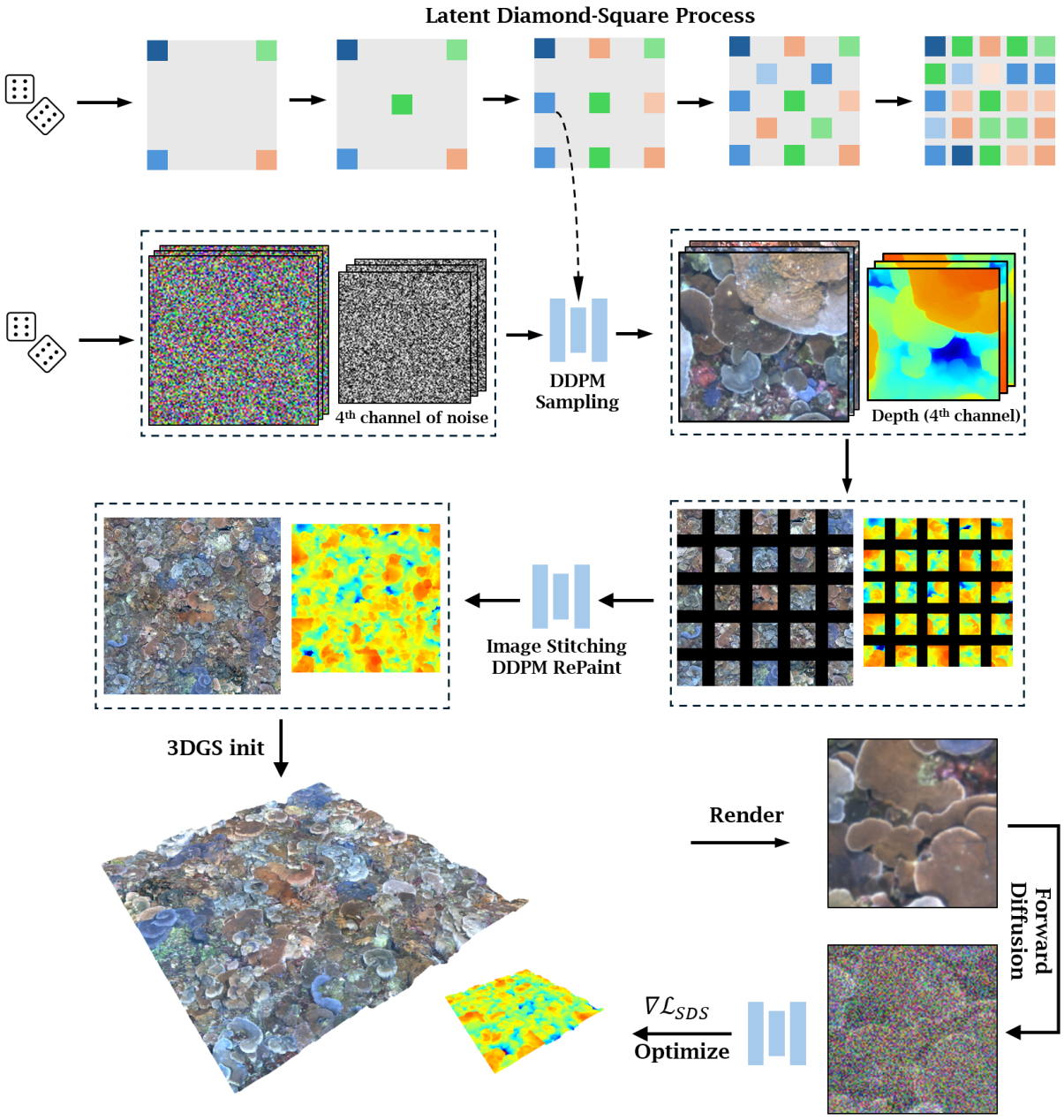


Figure 7.4: **Overview of Generation:** Our approach generates fractal embedding with the diamond-square method first, then generates images conditioned on these embeddings. We use RePaint [57] to stitch the images together into a dense RGBD map. The RGBD map can be converted into a 3D point cloud and initialized as a 3DGS model [49]. The 3DGS model is further refined with 2D diffusion priors using SDS loss allowing realistic rendering from novel views.

Let us denote the feature vector as

$$\boldsymbol{\phi} \leftarrow \texttt{PCA}(\texttt{DINOv2}(\mathbf{I})), \tag{7.1}$$

where $\mathbf{I}$ is an image and $\texttt{PCA}(\texttt{DINOv2}(\cdot))$ indicates applying PCA to the feature vector outputted by the DINO model, reducing dimensionality. During inference, our conditional generative model can be expressed as, $\mathbf{I} \sim P(\mathbf{I}|\boldsymbol{\phi})$, where $\boldsymbol{\phi}$ is a visual feature vector we condition upon. Generating spatially-consistent and yet diverse landscape images, requires controlling the evolution of $\boldsymbol{\phi}$ over the spatial domain, which alters the generative distribution of the terrain.

7.3.3 Fractal Latent Terrain Generation

An inherent property of naturally-occurring terrains is that coordinate points that are close in geometric distance should have similar attributes. The spatial distribution of natural terrain is often modeled using fractal processes to approximate natural-looking variations. We imbue this inductive bias into DreamSea through a novel **fractal embeddings framework**, which assumes that the latent vectors over the spatial domain follow fractal processes.

We begin by initializing the latent vectors at the corners of an arbitrary square region for which we seek to generate terrain. We seek to sample a latent function $\boldsymbol{\Phi} : \mathbb{R}^2 \rightarrow \mathbb{R}^d$, where d is the dimensionality of the latent vector after PCA reduction. Specifically, $\boldsymbol{\Phi}(\cdot)$ outputs a latent vector $\boldsymbol{\phi}$ for a given coordinate (x, y), which can then be used to control the image generation.

The latent function can be seen as a sample from a fractal process, generated from the *Diamond-Square* Algorithm applied to estimate the function output over a dense grid that covers the desired region. Here, the outputs are estimated recursively through a recursive two step process. First, in the *diamond step* we estimate the function value at the spatial mid-points of each square regions using the four corners of each square - forming four new diamonds. Next, we apply a *square step*, to estimate the mid-points of diamond regions from the corner points of each diamond — forming squares that subdivided the original square. In each step, we compute the latent vector values at the centers of square and diamond shape patterns as the mean of the corner points of the regions plus some random noise. Let us denote the set of vertices of a square or diamond shape as the set K, and the center point of the square or diamond as $\mathbf{r}_c$, the latent vector value at the center is

given by

$$\mathbf{\Phi}(\mathbf{r}_c) = \frac{1}{|K|} \sum_{\mathbf{r} \in K} \mathbf{\Phi}(\mathbf{r}) + s\boldsymbol{\sigma}, \qquad \boldsymbol{\sigma} \sim \mathcal{N}(0, \mathbf{I}). \tag{7.2}$$

Here, s is a scaling factor that controls the variability of the landscape. This factor s is gradually decayed. Therefore, starting with latent vector values at the vertices of a square, we can recursively estimate latent vector values over the entire square region.

A single iteration of this process, along with illustrated vertices, is shown in Figure 7.5. The end result of this step is a 2D spatial field of latent fractal embeddings that can be used to conditionally generate a set of images with strong spatial dependency.

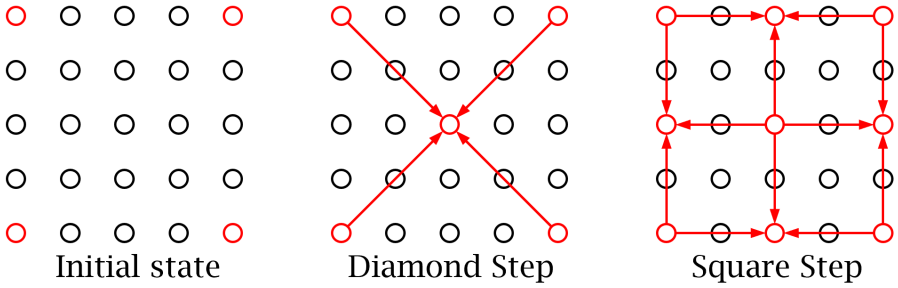


Figure 7.5: The Diamond-Square algorithm, which recursively interpolates on a spatial grid, is used to generate latent embeddings in our approach. The red arrows start from the vertices of the existing square and diamond shapes from the previous iteration, and point towards the new center points.

To accomplish this, we train a diffusion model using RGB images from real underwater imagery augmented with depth generated using Depth Anything v2 [122]. The resulting model is used to generate an RGBD image for each vertex in the spatial latent field and then RePaint [57] is used to in-fill any gaps between each pair of neighboring images, to form a spatially consistent map in the form of an RGBD point cloud.

Here, we highlight that the function of images over the 2D spatial domain is drawn from a *doubly stochastic process*. The set of generated images, $\{\mathbf{I}_\mathbf{x}\}_{\mathbf{x} \in \mathbb{R}^2}$, can be considered as a function drawn from the conditional diffusion model, which itself is dependent on a latent function, $\mathbf{\Phi}(\mathbf{x})$, drawn from a fractal process, governed by the scale factor s. Specifically,

$$\{\mathbf{I}_\mathbf{x}\}_{\mathbf{x} \in \mathbb{R}^2} \sim \underbrace{P(\mathbf{I}|\mathbf{\Phi}(\mathbf{x}))}_{\text{Diffusion Model}}, \qquad \mathbf{\Phi}(\mathbf{x}) \sim \underbrace{P(\mathbf{\Phi}|s)}_{\text{Fractal Process}}. \tag{7.3}$$

We note that the doubly stochastic nature of our image generation enables highly diverse terrains to be generated.

7.3.4 3D Scene Generation via Gaussian Splatting

In this section, we convert the RGBD point cloud generated in the previous step into a geometrically-consistent 3DGS model that uses the generated images as a strong prior. The resulting model provides us with a 3D structure that is dense and allows for the generation of novel images from arbitrary viewing poses.

We begin by using the depth channels from the generated images to initialize 3D Gaussians following the default method [49]. Then we freeze the 3D positions of the Gaussian cloud and refine the appearance with 2D diffusion priors. Given a cloud of Gaussians $\mathbf{G}$ initialized, each Gaussian g_i includes the following attributes: position $\mathbf{p}_i$, covariance Σ_i, opacity α_i and radiance $\mathbf{c}_i$, that $g_i = \{\mathbf{p}_i, \Sigma_i, \alpha_i, \mathbf{c}_i\} \in \mathbf{G}$. With a subset of Gaussians $\mathcal{N} \in G$ ordered along a camera ray, the pixel value in an image can be rendered from 3DGS models with the following rendering equation:

$$C = \sum_{i \in \mathcal{N}} \mathbf{c}_i \alpha_i \prod_{j=1}^{i-1} (1 - \alpha_j) \tag{7.4}$$

Here $\mathbf{p}_i$ is initialized from the depths of the generated images. We use the *Score Distillation Sampling* (SDS) loss introduced in DreamFusion [82] to optimize the 3D Gaussian model from 2D diffusion prior:

$$\nabla_\theta \mathcal{L}_{\text{SDS}}(\mathbf{I}^r) \triangleq \mathbb{E}_{t,\epsilon} \left[w(t) \left(\hat{\epsilon}(t) - \epsilon \right) \frac{\partial \mathbf{I}^r}{\partial \theta} \right] \tag{7.5}$$

here θ is the parameters of Gaussian cloud $\mathbf{G}$ to be optimized, $\mathbf{I}^r$ is the rendered image; $\hat{\epsilon}$ and ϵ are predicted noise and added noise; t is the timestep in the diffusion process and $w(t)$ is the weighting function following the implementation in [82] (parameter y and $\mathbf{z}_t$ in the original paper are omitted here for brevity).

7.4 Experiments

7.4.1 Datasets

The results presented throughout the chapter are trained on real-world data collected from four different locations with three different robot platforms, spanning a time from 2009 to 2024 (see Figure. 7.6). The *Scott Reef* and *Batemans datasets* were collected from 2009 to 2015 with a Seabed-class AUV, Sirius, which features a dual-hull design for stabilized imaging underwater. We post-process the raw images, hosted on Squidle.org, to obtain a normal exposure. The *Hawaii dataset* was collected in April 2024 with an Iver AUV, the torpedo design allowed it to travel long distances and sample images from the seafloor. The *Florida dataset* was collected in August 2023 with a customized ROV equipped with ZED cameras. Each location presents a unique benthic appearance and is reflected in our model.

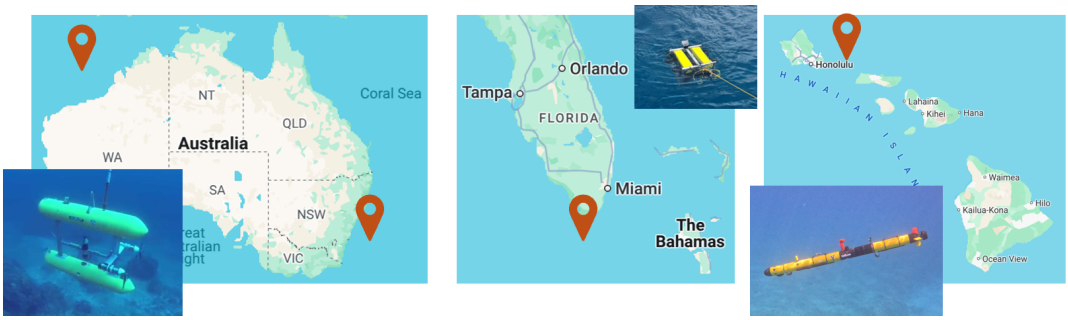


Figure 7.6: Results demonstrated in this chapter are trained on data collected from 4 different sites with 3 different robot platforms.

7.4.2 Implementation Details

Our model's implementation is adapted from DDPM networks. We train each model on a single NVIDIA RTX4090 GPU with 24GB VRAM for 2000 epochs, with a batch size of 12. Although the size of each data set differs, it usually takes $\sim$ 200 hours to train on a dataset with 10k images, at the resolution of 224×224. We use the first two main components from PCA results on DINO v2 embeddings. From our empirical study, we find it to be sufficient to describe the variation in appearance of underwater environments. This is consistent with the practice in [85, 126].

7.4.3 Qualitative Evaluation

We train the diffusion model on the dataset collected from various locations capturing diverse underwater appearances. At a glance, the generated images closely resemble the real images from the training set well, as shown in Figure 7.7. The generated relative depth also aligns well with human perception, indicating that our training pipeline successfully learns the visual distribution of real underwater datasets and distills the 3D information from the depth foundation model. This model that generates realistic RGB-D data serves as the cornerstone of the rest of this work.

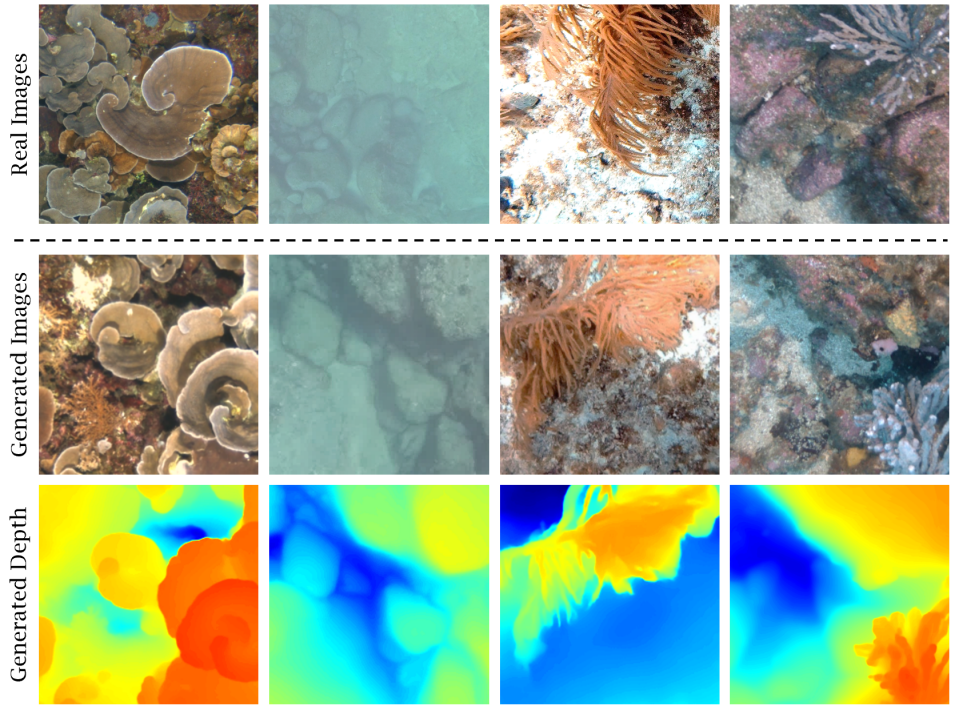


Figure 7.7: Our diffusion model is able to output realistic images as well as depth estimation distilled from depth anything v2 [122].

7.4.4 Image stitching by inpainting

Given two generated images spatially adjacent to each other, we stitch them together with RePaint [57]. Within the RePaint model, we investigate two approaches: 1) using the same conditional DDPM network used for generation; 2) training a new unconditional DDPM. The result shows that both methods can accomplish inpainting on the generated images. However, the conditional

inpainting model creates heavier boundary effects in the image, while unconditional inpainting creates fewer artifacts, as shown in Figure 7.9. Our hypothesis on this observation is that, for the conditioned inpaint approach, the neural network inpaints the image conditioned on both the existing part of the image as well as the latent embedding. Although they are sampled conditioned on the same latent embeddings, the actual appearance of the existing part may be shifted, creating inconsistencies when inpainting. The unconditional approach depends on the existing part of the image, so fewer artifacts are exhibited at the boundaries between images. The final results we present integrate an unconditional model to blend the images together, alongside the conditional image generation model.

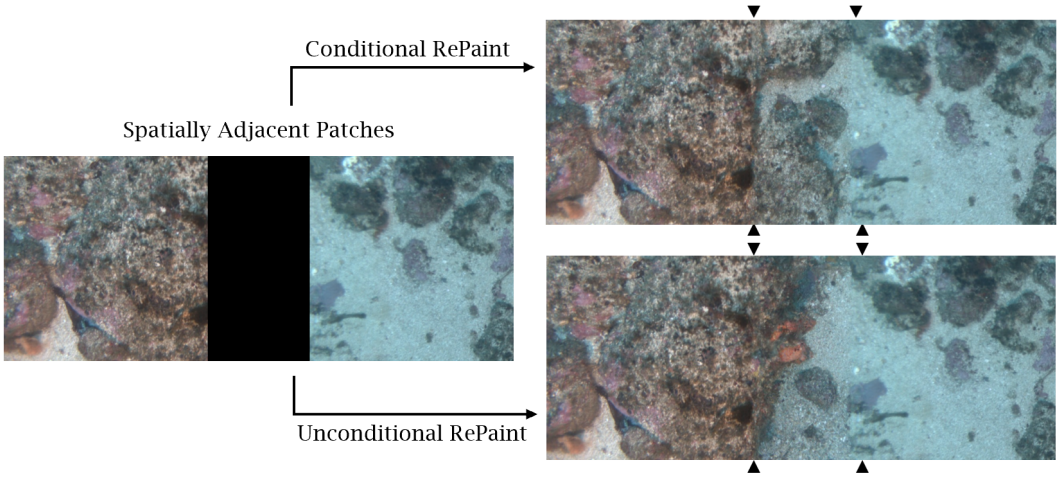


Figure 7.8: We find conditional repaint generates heavier boundary effects than unconditional repaint when blending images together.

7.4.5 Latent Controlled Generation

Generating images and maps with latent embedding control plays a critical role in creating terrain with appearance aligned with human preference and natural variation. We demonstrate a smooth image transition over the latent space: Figure 7.11 shows images generated with latent embedding interpolated in a 2D space. We can see how the appearance of the images smoothly transits along both axes and we can recognize how the content of the image shifts from reefs to sands to corals of different kinds. More results are shown in Figure 7.10 with diverse underwater scenes of different locations, which demonstrated that latent embeddings from VFMs controls underwater image generation smoothly and can be well aligned with human perception.

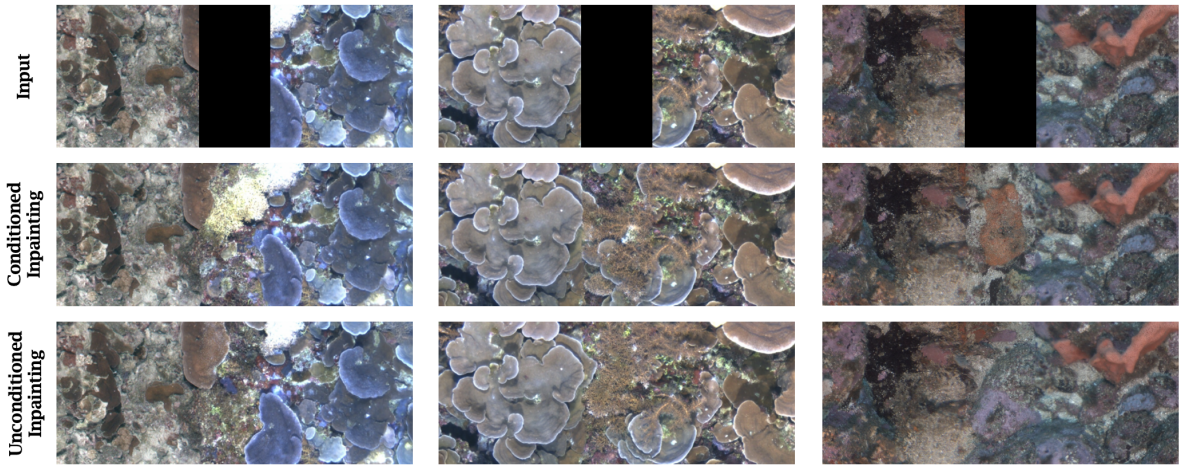


Figure 7.9: More visualization on conditional inpainting and unconditional inpainting: unconditional inpainting using RePaint shows the least boundry effects.

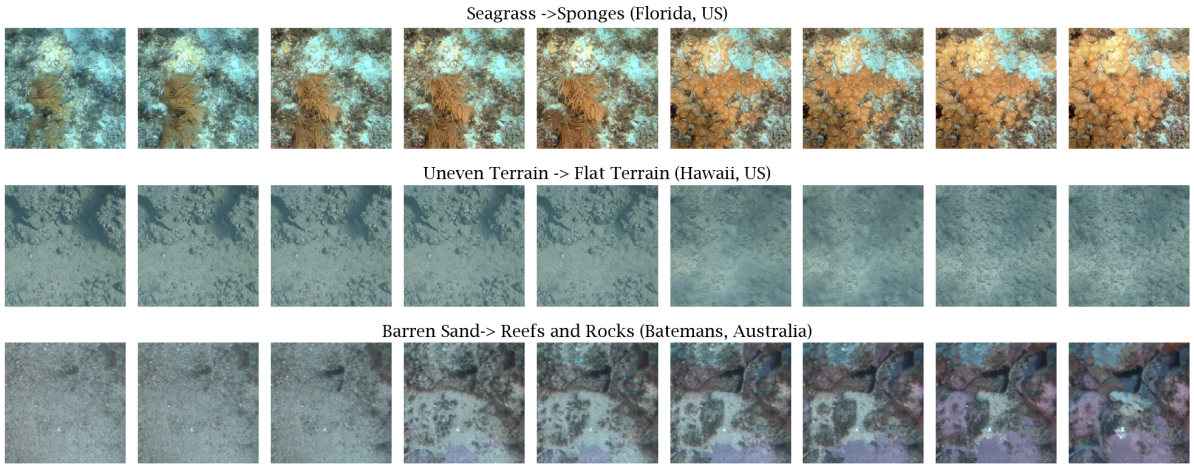


Figure 7.10: Examples of image generation conditioned on interpolated DINO embeddings. A smooth transition can be observed.

We further show the 2D map generated from a fractal latent field. In Figure 7.12, where the latent field is generated with $s = 0.6$, we observe that the stochasticity injected into the latent process visibly enhances the diversity of the generated terrain. We observe diverse patterns and elevations even when considering a local region. This locally diversity can be governed by tuning the scale factor s, further motivating our doubly stochastic formulation.

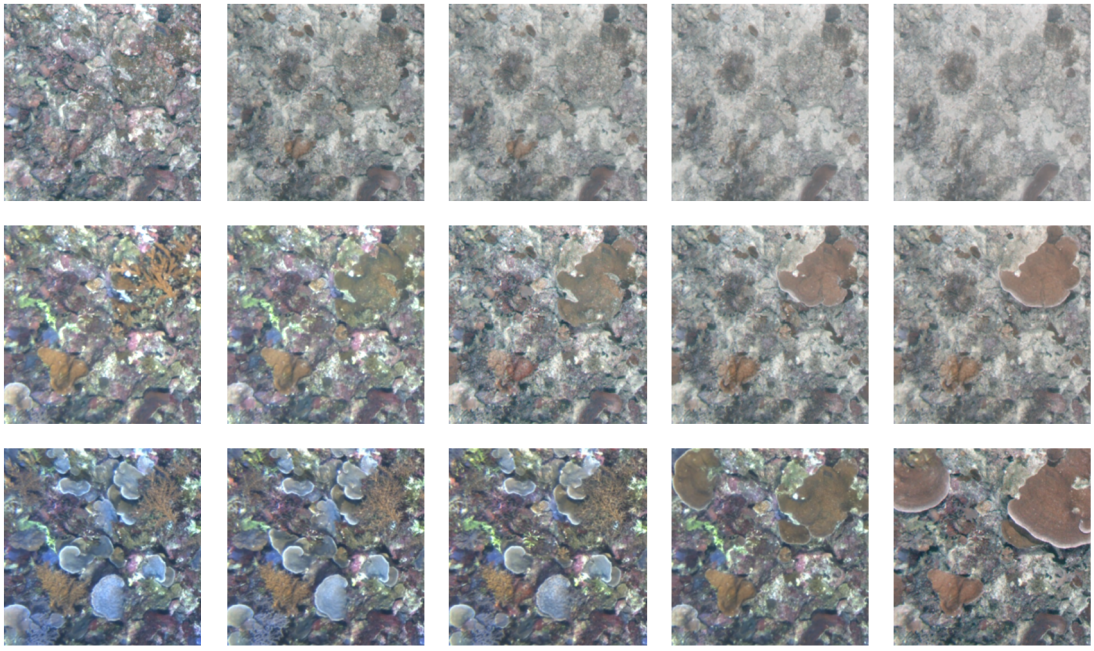


Figure 7.11: Interpolating on 2D latent space: we generate diverse images conditioned latent embeddings interpolated in 2 directions, and can observe the appearance of generated images gradually transitioning from sand to reef to corals of different kinds.

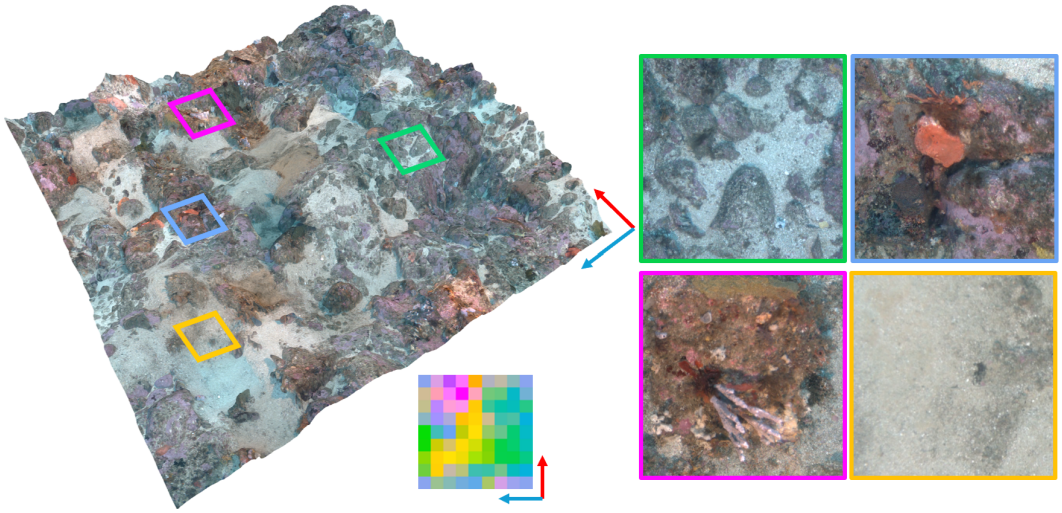


Figure 7.12: Latent Controlled Generation on fractal embeddings, with $s = 0.6$. Diversity observed even locally.

7.4.6 Inpainting Patterns

We further compare our inpainting pattern with most intuitive and commonly used patterns, i.e. raster scan pattern [53] and lawn mowing pattern [47]. The raster scan pattern updates the image space row by row in one direction. The lawn mowing pattern updates the image space row by row but in alternating direction, which is commonly used in robot mapping [47]. In comparison, the inpaint method introduced in this chapter is parallizable since the new patches are less dependent on previous generated patches. Furthermore, we demonstrate that such dependency reduces latent control accuracy by evaluating the CLIP and DINO latent of generated image patches (Reference embedding of DINO is given; for CLIP embedding we generate a batch of reference image and extract the CLIP embedding as reference). As shown in Table 7.1, which tabulates MSE between reference latent and predicted latent. Image patches are generated conditioned on input latent. We observe that by leveraging fractal embeddings, DreamSea consistently outperforms baselines that utilize raster scan and lawn mowing patterns which are sequential. These sequential in-painting patterns implicitly assume that the generated terrain contains auto-regressive dependencies while our fractal embeddings explicitly accounts for spatial dependencies along both x and y-axes.

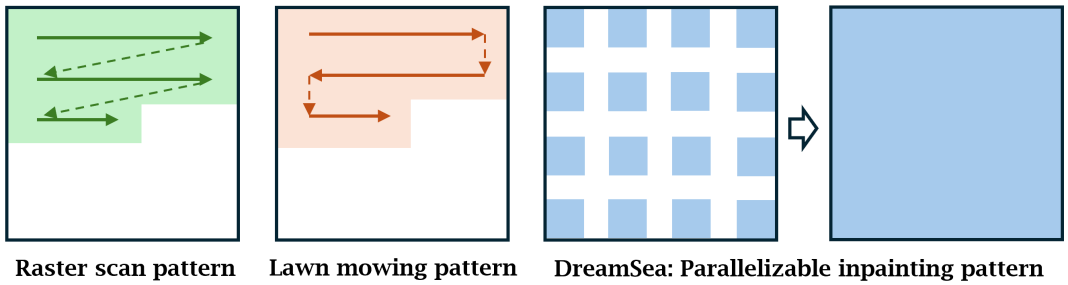


Figure 7.13: Our inpainting pattern is parallelizable, comparing to common patterns in image generation and robot mapping, i.e. raster pattern [53] and lawn mowing pattern [47].

Table 7.1: MSE↓ on CLIP [86]/DINO [76] embedding space evaluated on individual dataset Florida (FL), Hawaii (HI), Batemans (BM) and Scott Reef (SR). DreamSea outperforms as it does not generate images in a sequentially conditioned order.

	FL	HI	BM	SR	Ave.
Raster Order [53]	0.055/**3.44**	0.049/3.63	0.039/3.66	0.055/5.34	0.049/4.02
Lawn Mowing [47]	0.054/3.65	0.053/3.34	0.043/4.77	0.066/5.28	0.061/4.24
DreamSea	**0.035**/3.46	**0.029/2.12**	**0.030/2.95**	**0.041/4.48**	**0.034/3.34**

7.4.7 Ablation study: scaling factor s and damping factor ds

In the fractal process described in 7.3.3, two important hyperparameter are the scaling factor s and the damping factor ds. s determines the magnitude of randomness applied to the latents and ds is the factor that dampens s over iterations. Figure 7.14 shows how different values of s and ds affect the generated 3D terrain. With a small s (Figure 7.14 left), the latent map is smooth. Conditioned on this smooth latent mat which generates a 3D terrain with low variance in appearance. With large s (Figure 7.14 mid and right), spatial variance is observed and ds controls transition smoothness.

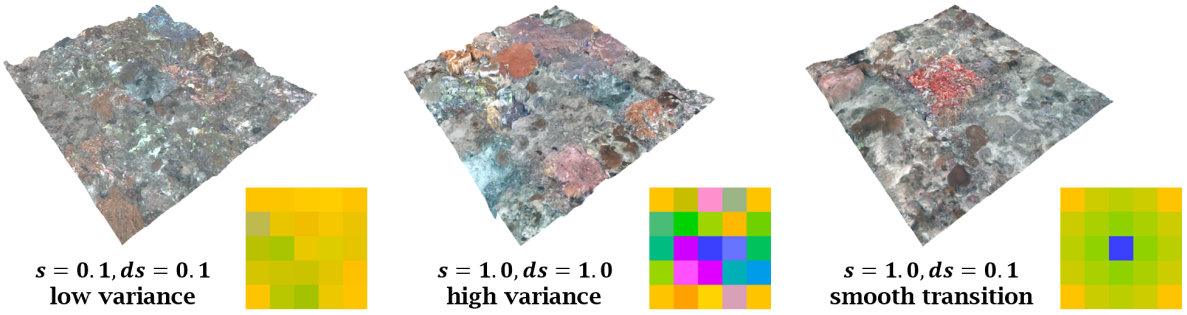


Figure 7.14: Effects of scaling factor s and dampening factor ds in fractal process: higher s yields higher variance in generated appearance, and lower ds will smooth the generate scene.

7.4.8 Towards underwater simulation environment

An example of the RGB map as well as the elevation is presented in Figure 7.15, both of which can be important in building a simulation pipeline for underwater perception and navigation. To better approximate the real world visual perturbations, we show that water effects [127, 130] and lighting effects [120, 129] studied in previous studies can be synthesized into our map, creating more realistic appearance for image rendering.

7.5 Limitations and Opportunities

Our current model only estimates relative as opposed to metric scale. The metric scale could optionally be acquired by auxiliary sensors such as IMUs, calibrated cameras, calibration targets, or single/multi-beam acoustic sensors.

Viewing angles are only from the top down. Although the datasets we use are from different institutes collected with different robot platforms, they are all from top-down view. This is

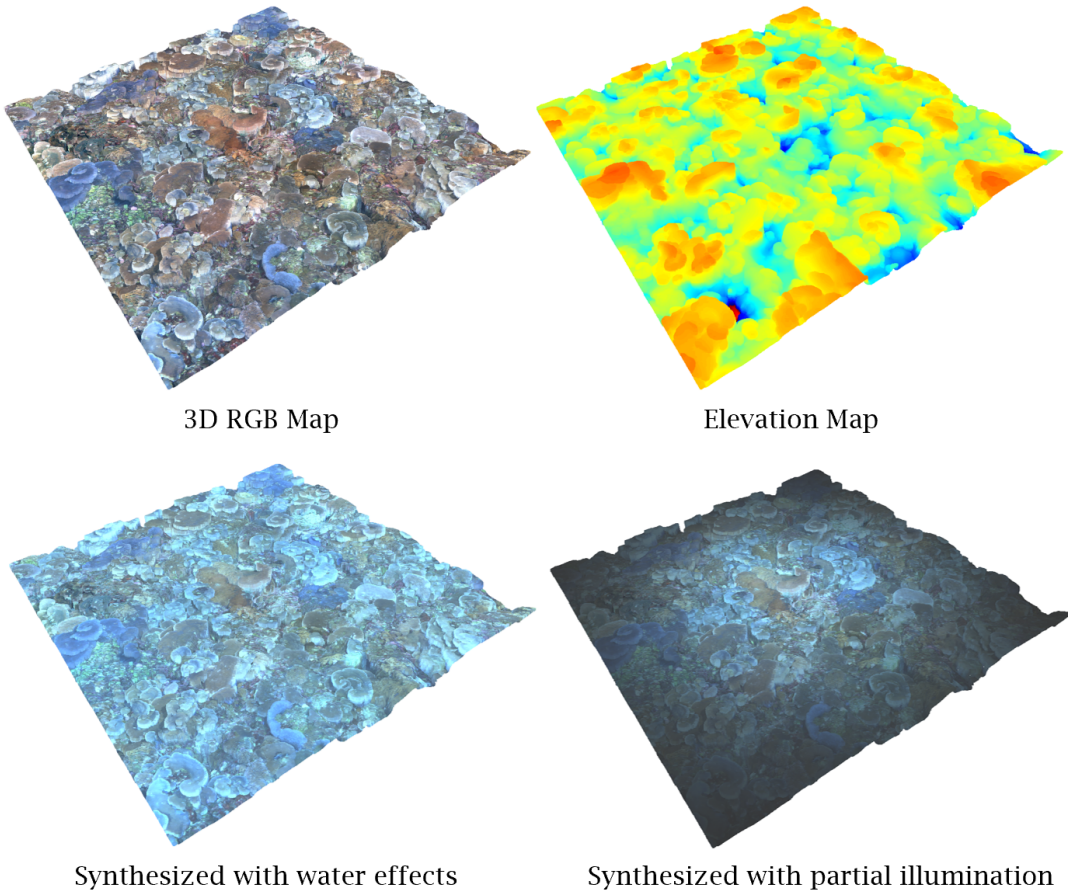


Figure 7.15: Elevation map, water effects and lighting effects can be integrated seamlessly to create realistic renderings.

constrained by the fact that each robot is designed to be passively stable in a hydrodynamic environment. This work further motivates new robot and perception system designs to allow for more diverse viewing angles.

It will also be useful to generate images which can integrate partial expert annotations to semi-supervise DreamSea. Determining how to bridge such a system with broader marine science, biography and geography community is still an open problem.

7.6 Conclusion

Generating realistic and diverse underwater terrains and scene representations has a wide variety of applications, spanning video games, movies, robotics, and marine science. Existing generative

methods struggle to generate sufficiently varied and physically accurate underwater images. To tackle this, we introduce *DreamSea*, a diffusion-based generative model which we train on a collection of large-scale unannotated underwater imagery collected by robots at different locations. Our approach conditions generation upon visual latent embeddings extracted using foundation models. Furthermore, DreamSea imbues spatial-awareness into the generative model via a novel fractal embedding algorithm. The resulting terrain generation allows for the generation of highly diverse underwater environments, while considering spatial-dependencies. The resulting terrain visuals and estimated depths are integrated as priors to construct 3D Gaussian Splatting models, which provide both 3D geometry and enables novel-view images to be produced. DreamSea is rigoriously evaluated and demonstrates the capability to generate large-scale hyper-realistic underwater scenes.

Chapter 8

Conclusions

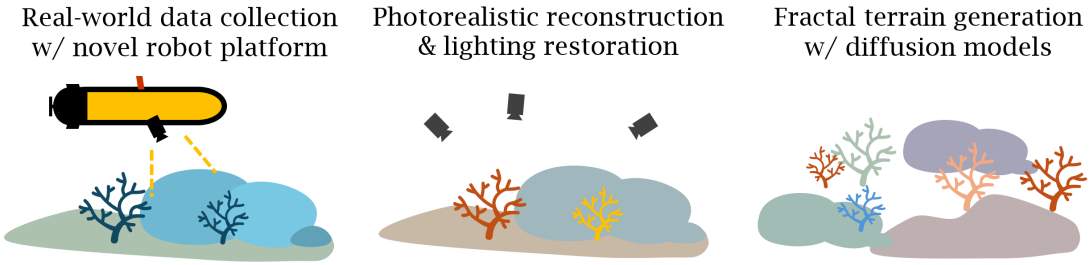


Figure 8.1: With real-world data collected by underwater robots, this dissertation introduce contributions that 1) reconstruct photorealistic 3D model of underwater scene 2) understand the complex and dynamic lighting effects and 3) create realistic 3D scenes with deep generative models.

This dissertation presents novel methodologies for reconstructing and generating 3D underwater environment. Ch. 1 and Ch. 2 introduced underwater 3D perception problems from a robotic perspective, and their close relation to well-studied computer vision techniques such as SfM and diffusion models. Ch. 3 and Ch. 4 present methodologies that build 3D representations under environmental light effects underwater, i.e. caustics and color distortions. Ch. 5 and Ch. 6 study the case that onboard camera moves with the onboard light source as a rigid body, which is not limited to underwater but all kinds of robot and autonomous systems working in the dark. With this setup, the robot will observe inconsistent illumination in the scene as the robot moves. The proposed method can calibrate the camera-light system and build 3D representation allowing simulating the lighting effects and recovering normal illumination. With large amount of data collected by

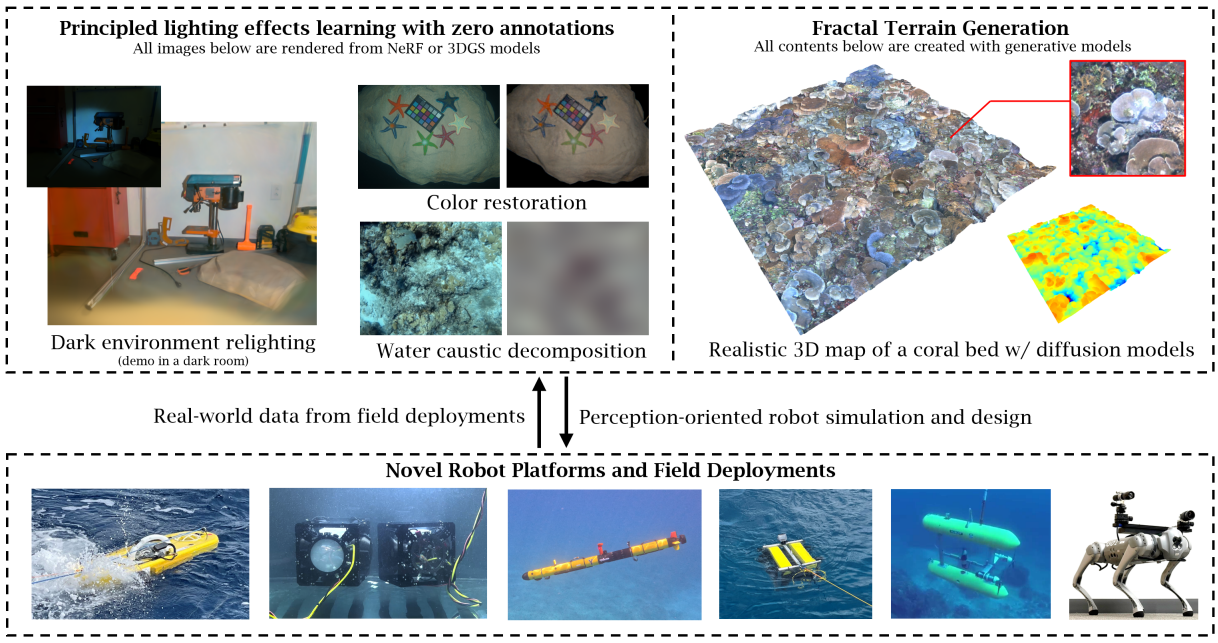


Figure 8.2: Driven by the real world data collected from field robot deployments, my research enables learning the physics of underwater lighting [127, 128, 130] and photorealistic 3D reconstruction under complex and dynamic illumination [120, 129]. We also investigated how deep generative models can be trained on robotic data to create realistic 3D underwater scenes, which can be used to simulate robot perception and feed back to novel underwater robot design [132].

underwater robots, Ch. 7 proposed a pipeline that trains 2D diffusion model to generate realistic seafloor terrain with 3D structure. The proposed diffusion model is controlled by latent fractals to present natural variance. It also shows that the water and lighting effects studied in previous chapters can be seamlessly applied to the generated 3D terrain, which has a significant implication to robot simulation.

This dissertation also presents major limitations and suggests following future research directions;

1. **Novel Robot Platform Design**: The 3D model created in this dissertation does not generalize well when viewing from side views. One reason is that the roll and pitch angle is limited in most underwater robots due to hydrostatic design. Underwater robots are usually designed to be passively stable for better maneuverability and imaging quality, but it turns out to be a double-edged sword that limits the diversity of viewing angles. The results in this dissertation imply that if the robot can be equipped with full 6-Degrees-of-Freedom mobility, the 3D reconstruction quality can be significantly improved.

2. **Active Perception and Imaging Control**: Underwater images often present low quality because of unknown underwater conditions, e.g. albedos, turbidity, or ambient lights. Current practice relies on human expertise in the design and deployment loop, which is usually inefficient and results in low-quality data with lost information. This suggests robot and camera control algorithms that proactively adapt to the environment to collect better images.
3. **Underwater Simulators with Photorealistic Quality**: Visual perception is expected to play an increasingly important role in high-resolution seafloor mapping in the near future. However, most of today's underwater simulators do not support photorealistic image simulation. Insights from the broader robotics community have shown that photorealistic simulation is crucial for bridging the sim-to-real gap when training deep learning models. Developing open-source underwater simulators with improved rendering quality and support for diverse 3D structures would provide valuable benefits to the community.

Together, the contributions and lessons learned in this dissertation closed the loop of data collection from field robot deployments, 3D reconstruction underwater real-world lighting perturbations, and generating 3D terrains that present a natural appearance, which can be potentially used for robot simulation. It shows how state-of-the-art photorealistic 3D representations and generative models can be adapted to field robot data and integrated with physics laws, statistical principles, and foundation models, without involving any human annotations.

Bibliography

[1] Image technology colour management-architecture, profile format and data structure-part 1: Based on icc.1:2010. Technical Report 15076-1, ISO, 2010. URL https://www.iso.org/standard/54754.html. 4.4.3

[2] Adobe Systems Incorporated. Inverting the color component transfer function. https://www.adobe.com/digitalimag/pdfs/AdobeRGB1998.pdf. 4.3.6

[3] Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M. Seitz, and Rick Szeliski. Building rome in a day. *Communications of the ACM*, 54:105–112, 2011. (document), 2.1, 2.1.2, 3.1, 3.3.1, 6.1, 6.2.2, 7.1

[4] Ali Agha, Kyohei Otsu, Benjamin Morrell, David D Fan, Rohan Thakker, Angel Santamaria-Navarro, Sung-Kyun Kim, Amanda Bouman, Xianmei Lei, Jeffrey Edlund, et al. Nebula: Quest for robotic autonomy in challenging environments; team costar at the darpa subterranean challenge. *arXiv preprint arXiv:2103.11470*, 2021. (document), 6.1

[5] Panagiotis Agrafiotis, Konstantinos Karantzalos, and Andreas Georgopoulos. Seafloor-invariant caustics removal from underwater imagery. *IEEE Journal of Oceanic Engineering*, 48(4):1300–1321, 2023. doi: 10.1109/JOE.2023.3277168. 3.1, 3.2.2, 3.3.1, 3.4.3

[6] Derya Akkaynak and Tali Treibitz. A revised underwater image formation model. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6723–6732, 2018. doi: 10.1109/CVPR.2018.00703. 4.3.3

[7] Derya Akkaynak, Tali Treibitz, Tom Shlesinger, Yossi Loya, Raz Tamir, and David Iluz. What is the space of attenuation coefficients in underwater computer vision? In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 568–577, 2017. doi: 10.1109/CVPR.2017.68. 4.3.2

[8] Arthur J. Bachrach. History of diving. *Historical Diving Times*, 1998. 1.1

[9] Jonathan T. Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P. Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5835–5844, 2021. URL https://api.semanticscholar.org/CorpusID:232352655. 2.2.2

[10] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. *CVPR*, 2022. 4.2

[11] Gilad Baruch, Zhuoyuan Chen, Afshin Dehghan, Tal Dimry, Yuri Feigin, Peter Fu, Thomas Gebauer, Brandon Joffe, Daniel Kurz, Arik Schwartz, and Elad Shulman. ARKitscenes - a diverse real-world dataset for 3d indoor scene understanding using mobile RGB-d data. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021. URL https://openreview.net/forum?id=tjZjv_qh_CE. 2.1.4

[12] Ronen Basri, David Jacobs, and Ira Kemelmacher. Photometric stereo with general, unknown lighting. *International Journal of Computer Vision*, 72:239–257, 05 2007. doi: 10.1109/CVPR.2001.990985. 6.1

[13] Jorge Beltrán, Carlos Guindel, Arturo de la Escalera, and Fernando García. Automatic extrinsic calibration method for lidar and camera sensor setups. *IEEE Transactions on Intelligent Transportation Systems*, 2022. doi: 10.1109/TITS.2022.3155228. 5.2

[14] Sai Bi, Zexiang Xu, Pratul Srinivasan, Ben Mildenhall, Kalyan Sunkavalli, Miloš Hašan, Yannick Hold-Geoffroy, David Kriegman, and Ravi Ramamoorthi. Neural reflectance fields for appearance acquisition. *arXiv preprint arXiv:2008.03824*, 2020. 2.2.1, 4.3.1, 4.3.4

[15] Sai Bi, Zexiang Xu, Pratul Srinivasan, Ben Mildenhall, Kalyan Sunkavalli, Miloš Hašan, Yannick Hold-Geoffroy, David Kriegman, and Ravi Ramamoorthi. Neural reflectance fields for appearance acquisition. *arXiv preprint arXiv:2008.03824*, 2020. doi: 10.48550/ARXIV. 2008.03824. URL https://arxiv.org/abs/2008.03824. 1.2, 2.2.2, 4.1, 5.4.1

[16] Mitch Bryson, Matthew Johnson-Roberson, Oscar Pizarro, and Stefan B. Williams. True color correction of autonomous underwater vehicle imagery. *Journal of Field Robotics*, 33(6):853–874, 2016. doi: https://doi.org/10.1002/rob.21638. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21638. 4.1, 4.2, 4.3.2

[17] Gershon Buchsbaum. A spatial processor model for object colour perception. *Journal of the Franklin institute*, 310(1):1–26, 1980. 4.2, 4.4.3

[18] Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, José Neira, Ian Reid, and John J Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on robotics*, 32(6): 1309–1332, 2016. 2.1.3

[19] Shengze Cai, Zhiping Mao, Zhicheng Wang, Minglang Yin, and George Em Karniadakis. Physics-informed neural networks (pinns) for fluid mechanics: A review. *Acta Mechanica Sinica*, 37(12):1727–1738, 2021. 3.2.2

[20] Carlos Campos, Richard Elvira, Juan J. Gómez, José M. M. Montiel, and Juan D. Tardós. ORB-SLAM3: An accurate open-source library for visual, visual-inertial and multi-map SLAM. *IEEE Transactions on Robotics*, 37(6):1874–1890, 2021. (document), 2.1, 2.1.3, 6.1

[21] Ayan Chakrabarti and Kalyan Sunkavalli. Single-image rgb photometric stereo with spatially-varying albedo. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 258–266. IEEE, 2016. 6.1

[22] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017. 2.1.4

[23] François Darmon, Lorenzo Porzi, Samuel Rota-Bulò, and Peter Kontschieder. Robust gaussian splatting, 2024. 3.1

[24] Matt Deitke, Eli VanderBilt, Alvaro Herrasti, Luca Weihs, Kiana Ehsani, Jordi Salvador, Winson Han, Eric Kolve, Aniruddha Kembhavi, and Roozbeh Mottaghi. Procthor: Large-scale embodied ai using procedural generation. *Advances in Neural Information Processing Systems*, 35:5982–5994, 2022. 7.2.1

[25] Joris Domhof, Julian F. P. Kooij, and Dariu M. Gavrila. A joint extrinsic calibration tool for radar, camera and lidar. *IEEE Transactions on Intelligent Vehicles*, 6(3):571–582, 2021. doi: 10.1109/TIV.2021.3065208. 5.2

[26] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017. (document), 6.1

[27] C Edmonds, C Lowry, and J Pennefather. History of diving. *Journal of the South Pacific Underwater Medicine Society*, 1975. 1.1

[28] Jean Feydy, Thibault Séjourné, François-Xavier Vialard, Shun-ichi Amari, Alain Trouve, and Gabriel Peyré. Interpolating between optimal transport and mmd using sinkhorn divergences. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2681–2690, 2019. 4.4.3

[29] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, 6 1981. ISSN 0001-0782. doi: 10.1145/358669.358692. URL https://doi.org/10.1145/358669.358692. 5.3

[30] Julian Fong, Magnus Wrenninge, Christopher Kulla, and Ralf Habel. Production volume rendering: Siggraph 2017 course. In *ACM SIGGRAPH 2017 Courses*, pages 1–79. 2017. 4.3.2, 4.3.2

[31] Timothy Forbes, Mark Goldsmith, Sudhir Mudur, and Charalambos Poullis. Deepcaustics: Classification and removal of caustics from underwater imagery. *IEEE Journal of Oceanic Engineering*, 44(3):728–738, 2019. doi: 10.1109/JOE.2018.2838939. 3.1, 3.2.2, 3.3.1

[32] Alain Fournier, Don Fussell, and Loren Carpenter. *Computer rendering of stochastic models*, page 189–202. Association for Computing Machinery, New York, NY, USA, 1998. ISBN 158113052X. URL https://doi.org/10.1145/280811.280993. 7.2.1

[33] Jian Gao, Chun Gu, Youtian Lin, Hao Zhu, Xun Cao, Li Zhang, and Yao Yao. Relightable 3d gaussian: Real-time point cloud relighting with brdf decomposition and ray tracing. *arXiv:2311.16043*, 2023. (document), 2.2.2, 6.1, 6.4, 3, 6.3.1

[34] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361, 2012. doi: 10.1109/CVPR.2012.6248074. 2.1.4

[35] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020. 2.3.1, 7.2.2

[36] Nuno Gracias, Shahriar Negahdaripour, Laszlo Neumann, Ricard Prados, and Rafael Garcia. A motion compensated filtering approach to remove sunlight flicker in shallow water images. In *OCEANS 2008*, pages 1–7, 2008. doi: 10.1109/OCEANS.2008.5152111. (document), 3.1, 3.2.1, 3.3.1, 3.3.2, 3.7

[37] Klaus Greff, Francois Belletti, Lucas Beyer, Carl Doersch, Yilun Du, Daniel Duckworth, David J Fleet, Dan Gnanapragasam, Florian Golemo, Charles Herrmann, et al. Kubric: A scalable dataset generator. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3749–3761, 2022. 7.2.1

[38] Richard Hartley. *Multiple view geometry in computer vision*, volume 665. Cambridge university press, 2003. (document), 2.1, 2.1.2

[39] Kaiming He, Jian Sun, and Xiaoou Tang. Single image haze removal using dark channel prior. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1956–1963, 2009. doi: 10.1109/CVPR.2009.5206515. 4.2

[40] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 2.3.1, 7.2.2

[41] Woods Hole Oceanographic Institution. History of alvin, Jan 2019. URL https://www.whoi.edu/what-we-do/explore/underwater-vehicles/hov-alvin/history-of-alvin/. 1.1

[42] Md Jahidul Islam, Youya Xia, and Junaed Sattar. Fast underwater image enhancement for improved visual perception. *IEEE Robotics and Automation Letters*, 5(2):3227–3234, 2020. doi: 10.1109/LRA.2020.2974710. 4.1, 4.1, 4.2, 4.4.3

[43] J.S. Jaffe. Computer modeling and the design of optimal underwater imaging systems. *IEEE Journal of Oceanic Engineering*, 15(2):101–111, 1990. doi: 10.1109/48.50695. 4.1, 4.1, 4.2, 4.3.2, 4.3.3

[44] Nils Gunnar Jerlov. *Marine optics*. Elsevier, 1976. 4.3.3, 4.4.1

[45] Yingwenqi Jiang, Jiadong Tu, Yuan Liu, Xifeng Gao, Xiaoxiao Long, Wenping Wang, and Yuexin Ma. Gaussianshader: 3d gaussian splatting with shading functions for reflective surfaces. *arXiv preprint arXiv:2311.17977*, 2023. 2.2.2

[46] Wei Jin, Mingzhu Zhu, Jiantao Liu, Bingwei He, and Junzhi Yu. Shadow-based lightsource localization with direct camera–lightsource geometry. *IEEE Transactions on Instrumentation and Measurement*, 73:1–12, 2024. doi: 10.1109/TIM.2023.3344148. 5.2

[47] Matthew Johnson-Roberson, Oscar Pizarro, Stefan B. Williams, and Ian Mahon. Generation and visualization of large-scale three-dimensional reconstructions from underwater robotic surveys. *Journal of Field Robotics*, 27(1):21–51, 2010. doi: https://doi.org/10.1002/rob.20324. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.20324. (document), 7.4.6, 7.13, 7.1

[48] R. Karlsson and F. Gustafsson. Particle filter for underwater terrain navigation. In *IEEE Workshop on Statistical Signal Processing, 2003*, pages 526–529, 2003. doi: 10.1109/SSP.2003.1289507. 2.1.3

[49] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), July 2023. URL https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/. (document), 1.2, 1.2, 2.2, 2.2.2, 3.1, 3.3.1, 3.3.1, 3.4.1, 6.1, 6.2.1, 1, 6.4, 6.3.1, 3, 7.4, 7.3.4

[50] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12 2014. 5.4.1, 6.3

[51] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *stat*, 1050:1, 2014. 2.3.1, 7.2.2, 7.3.2

[52] Jie Li, Katherine A. Skinner, Ryan M. Eustice, and Matthew Johnson-Roberson. Watergan: Unsupervised generative network to enable real-time color correction of monocular underwater images. *IEEE Robotics and Automation Letters*, 3:387–394, 2017. 4.1, 4.2

[53] Tianhong Li, Yonglong Tian, He Li, Mingyang Deng, and Kaiming He. Autoregressive image generation without vector quantization. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 56424–56445. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/66e226469f20625aaebddbe47f0ca997-Paper-Conference.pdf. (document), 7.4.6, 7.13, 7.1

[54] Fayao Liu, Chunhua Shen, and Guosheng Lin. Deep convolutional neural fields for depth estimation from a single image. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5162–5170, 2015. doi: 10.1109/CVPR.2015.7299152. 2.1.4

[55] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60:91–110, 2004. 2.1.1, 6.1

[56] Bruce D Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *IJCAI'81: 7th international joint conference on Artificial intelligence*, volume 2, pages 674–679, 1981. 2.1.1

[57] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11461–11471, June 2022. (document), 2.3.1, 7.2.2, 7.4, 7.3.3, 7.4.4

[58] Long Ma, Jirui Liu, Xin Pei, Yanmin Hu, and Fengming Sun. Calibration of position and orientation for point light source synchronously with single image in photometric stereo. *Opt. Express*, 27(4):4024–4033, 2 2019. doi: 10.1364/OE.27.004024. URL https://opg.optica.org/oe/abstract.cfm?URI=oe-27-4-4024. 5.2, 5.3.1, 5.4.3

[59] Long Ma, Yuzhe Liu, Jirui Liu, Shengwei Guo, Xin Pei, Fengming Sun, and Shaobo Fang. A fast led calibration method under near field lighting based on photometric stereo. *Optics and Lasers in Engineering*, 147:106749, 2021. ISSN 0143-8166. doi: https://doi.org/10.1016/j.optlaseng.2021.106749. URL https://www.sciencedirect.com/science/article/pii/S0143816621002190. 5.2

[60] Long Ma, Xu Liu, Yuzhe Liu, Xin Pei, and Shengwei Guo. Robust point light source calibration method for near-field photometric stereo using feature points selection. *Appl. Opt.*, 62(36):9512–9522, 12 2023. doi: 10.1364/AO.505234. URL https://opg.optica.org/ao/abstract.cfm?URI=ao-62-36-9512. 5.2, 5.3.1, 5.4.2, 5.4.3

[61] B.B. Mandelbrot. *The Fractal Geometry of Nature*. Einaudi paperbacks. Henry Holt and Company, 1983. ISBN 9780716711865. URL https://books.google.com/books?id=0R2LkE3N7-oC. 7.2.1

[62] Travis Manderson, Jimmy Li, Natasha Dudek, David Meger, and Gregory Dudek. Robotic coral reef health assessment using automated image analysis. *Journal of Field Robotics*, 34(1):170–187, 2017. doi: https://doi.org/10.1002/rob.21698. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21698. 4.1

[63] N. Max. Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):99–108, 1995. doi: 10.1109/2945.468400. 4.3.5

[64] Calvin S. McCamy, H. Marcus, and J. G. Davidson. A color-rendition chart. *J. Appl. Photogr. Eng.*, 2(3):95–99, Summer 1976. URL http://scitation.aip.org/getabs/insref_abs.jsp?key=JEIME5&prog=getinsref&id=983288&idtype=inspec. 4.4.1

[65] B. L. McGlamery. A Computer Model For Underwater Camera Systems. In Seibert Quimby Duntley, editor, *Ocean Optics VI*, volume 0208, pages 221 – 231. International Society for Optics and Photonics, SPIE, 1980. doi: 10.1117/12.958279. URL https://doi.org/10.1117/12.958279. 4.1, 4.2

[66] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 2.2.1, 3.1, 4.3.5, 6.1

[67] 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.

Commun. ACM, 65(1):99–106, December 2021. ISSN 0001-0782. doi: 10.1145/3503250. URL https://doi.org/10.1145/3503250. (document), 1.2, 1.2, 2.2

[68] Ben Mildenhall, Peter Hedman, Ricardo Martin-Brualla, Pratul P. Srinivasan, and Jonathan T. Barron. NeRF in the dark: High dynamic range view synthesis from noisy raw images. *CVPR*, 2022. (document), 4.2, 4.3.6, 6.1, 6.4, 2, 6.3.1

[69] Gavin S P Miller. The definition and rendering of terrain maps. In *Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '86, page 39–48, New York, NY, USA, 1986. Association for Computing Machinery. ISBN 0897911962. doi: 10.1145/15922.15890. URL https://doi.org/10.1145/15922.15890. 7.2.1

[70] Curtis D Mobley. *Light and water: radiative transfer in natural waters*. Academic press, 1994. 4.1

[71] Edward Morgan, William Ard, and Corina Barbalata. A probabilistic framework for hydrodynamic parameter estimation for underwater manipulators. In *OCEANS 2023 - MTS/IEEE U.S. Gulf Coast*, pages 1–9, 2023. doi: 10.23919/OCEANS52994.2023.10337120. (document), 3.4.1, 3.4

[72] Linus Mossberg. Monte Carlo Ray Tracer. https://github.com/linusmossberg/monte-carlo-ray-tracer, 2022. URL https://github.com/linusmossberg/monte-carlo-ray-tracer. 4.4.1

[73] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4): 102:1–102:15, July 2022. doi: 10.1145/3528223.3530127. URL https://doi.org/10.1145/3528223.3530127. 2.2.1, 2.2.2, 4.3.1, 4.4.2

[74] Raúl Mur-Artal, J. M. M. Montiel, and Juan D. Tardós. ORB-SLAM: a versatile and accurate monocular SLAM system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015. doi: 10.1109/TRO.2015.2463671. 6.1, 7.1

[75] Edwin Olson. Apriltag: A robust and flexible visual fiducial system. In *2011 IEEE International Conference on Robotics and Automation*, pages 3400–3407, 2011. doi: 10.1109/ICRA.2011.5979561. 5.1, 5.2, 5.2

[76] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=a68SUt6zFt. Featured Certification. (document), 7.2.3, 7.3, 7.3.2, 7.1

[77] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Pro-*

ceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pages 4195–4205, October 2023. 2.3.1, 7.2.2

[78] Ken Perlin. An image synthesizer. In *Proceedings of the 12th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '85, page 287–296, New York, NY, USA, 1985. Association for Computing Machinery. ISBN 0897911660. doi: 10.1145/325334.325247. URL https://doi.org/10.1145/325334.325247. 7.2.1

[79] Theodore J Petzold. Volume scattering functions for selected ocean waters. Technical report, Scripps Institution of Oceanography La Jolla Ca Visibility Lab, 1972. 4.1, 4.2, 4.4.1

[80] Matt Pharr, Wenzel Jakob, and Greg Humphreys. *Physically based rendering: From theory to implementation*. Morgan Kaufmann, 2016. 4.3.2

[81] Stephen M Pizer, E Philip Amburn, John D Austin, Robert Cromartie, Ari Geselowitz, Trey Greer, Bart ter Haar Romeny, John B Zimmerman, and Karel Zuiderveld. Adaptive histogram equalization and its variations. *Computer vision, graphics, and image processing*, 39(3):355–368, 1987. 4.2, 4.4.3

[82] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. In *The Eleventh International Conference on Learning Representations*. 2.3.2, 7.3.4, 7.3.4

[83] Robin M. Pope and Edward S. Fry. Absorption spectrum (380–700 nm) of pure water. ii. integrating cavity measurements. *Appl. Opt.*, 36(33):8710–8723, Nov 1997. doi: 10.1364/AO.36.008710. URL https://opg.optica.org/ao/abstract.cfm?URI=ao-36-33-8710. 4.1, 4.1

[84] Easton Potokar, Spencer Ashford, Michael Kaess, and Joshua G. Mangelson. Holoocean: An underwater robotics simulator. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 3040–3046, 2022. doi: 10.1109/ICRA46639.2022.9812353. (document), 6.1

[85] Minghan Qin, Wanhua Li, Jiawei Zhou, Haoqian Wang, and Hanspeter Pfister. Langsplat: 3d language gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20051–20060, 2024. 7.3.2, 7.4.2

[86] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. (document), 7.2.3, 7.3.2, 7.1

[87] Alexander Raistrick, Lahav Lipson, Zeyu Ma, Lingjie Mei, Mingzhe Wang, Yiming Zuo, Karhan Kayan, Hongyu Wen, Beining Han, Yihan Wang, Alejandro Newell, Hei Law, Ankit Goyal, Kaiyu Yang, and Jia Deng. Infinite photorealistic worlds using procedural generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12630–12641, 2023. 7.2.1

[88] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12159–12168, 2021. doi: 10.1109/ICCV48922.2021.01196. 2.1.4

[89] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3d deep learning with pytorch3d. *arXiv:2007.08501*, 2020. 4.4.2

[90] Joern Rehder, Janosch Nikolic, Thomas Schneider, Timo Hinzmann, and Roland Siegwart. Extending kalibr: Calibrating the extrinsics of multiple imus and of individual axes. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4304–4311. IEEE, 2016. 2.1.1, 5.2

[91] Jeremy Reizenstein, Roman Shapovalov, Philipp Henzler, Luca Sbordone, Patrick Labatut, and David Novotny. Common objects in 3d: Large-scale learning and evaluation of real-life 3d category reconstruction. In *International Conference on Computer Vision*, 2021. 2.1.4

[92] Maik Riechert. Rawpy. `https://pypi.org/project/rawpy/`, 2023. 4.4.1

[93] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. (document), 2.3.1, 7.2.2, 7.2

[94] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham, 2015. Springer International Publishing. ISBN 978-3-319-24574-4. 1.2, 3.2.2

[95] A. Ryer. *Light Measurement Handbook*. International Light, 1997. ISBN 9780965835695. URL `https://books.google.com/books?id=e-JIPwAACAAJ`. (document), 5.3.2, 5.3.2, 5.4.3, 5.7

[96] Hiroaki Santo, Michael Waechter, Masaki Samejima, Yusuke Sugano, and Yasuyuki Matsushita. Light structure from pin motion: Simple and accurate point light calibration for physics-based modeling. In *European Conference on Computer Vision (ECCV)*, 2018. 5.2

[97] Hiroaki Santo, Michael Waechter, Wen-Yan Lin, Yusuke Sugano, and Yasuyuki Matsushita. Light structure from pin motion: Geometric point light source calibration. *International Journal of Computer Vision (IJCV)*, 2020. doi: 10.1007/s11263-020-01312-3. 5.2

[98] Y.Y. Schechner and N. Karpel. Attenuating natural flicker patterns. In *Oceans '04 MTS/IEEE Techno-Ocean '04 (IEEE Cat. No.04CH37600)*, 2004. 3.1, 3.2.1

[99] Y.Y. Schechner and N. Karpel. Recovery of underwater visibility and structure by polarization analysis. *IEEE Journal of Oceanic Engineering*, 30(3):570–587, 2005. doi: 10.1109/JOE.2005.850871. 4.2

[100] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2.1.2, 4.4.1, 6.1

[101] Advaith Venkatramanan Sethuraman, Manikandasriram Srinivasan Ramanagopal, and Katherine A. Skinner. Waternerf: Neural radiance fields for underwater scenes. *ArXiv*, abs/2209.13091, 2022. 4.1, 4.2, 4.4.3, 4.4.3

[102] Asm Shihavuddin, Nuno Gracias, and Rafael Garcia. Online sunflicker removal using dynamic texture prediction. *Proceedings of the International Conference on Computer Vision Theory and Applications*, 2012. 3.1, 3.2.1

[103] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part V 12*, pages 746–760. Springer, 2012. 2.1.4

[104] Hanumant Singh, Ali Can, Ryan Eustice, Steve Lerner, Neil McPhee, and Chris Roman. Seabed auv offers new platform for high-resolution imaging. *Eos, Transactions American Geophysical Union*, 85(31):289–296, 2004. 1.1

[105] Yifan Song, Furkan Elibol, Mengkun She, David Nakath, and Kevin Köser. Light pose calibration for camera-light vision systems, 2020. 5.2, 5.3.1, 5.4.2, 5.4.3

[106] Yifan Song, David Nakath, Mengkun She, Furkan Elibol, and Kevin Köser. Deep sea robotic imaging simulator. In Alberto Del Bimbo, Rita Cucchiara, Stan Sclaroff, Giovanni Maria Farinella, Tao Mei, Marco Bertini, Hugo Jair Escalante, and Roberto Vezzani, editors, *Pattern Recognition. ICPR International Workshops and Challenges*, pages 375–389, Cham, 2021. Springer International Publishing. ISBN 978-3-030-68790-8. 4.1, 4.2, 4.4.1

[107] Pratul P. Srinivasan, Boyang Deng, Xiuming Zhang, Matthew Tancik, Ben Mildenhall, and Jonathan T. Barron. Nerv: Neural reflectance and visibility fields for relighting and view synthesis. In *CVPR*, 2021. 5.3.4

[108] D Steinberg, Alon Friedman, Oscar Pizarro, and Stefan Williams. A bayesian nonparametric approach to clustering data from underwater robotic surveys. In *International Symposium on Robotics Research*, 01 2011. (document), 1.1, 1.3

[109] Jiaxiang Tang, Jiawei Ren, Hang Zhou, Ziwei Liu, and Gang Zeng. Dreamgaussian: Generative gaussian splatting for efficient 3d content creation. In *The Twelfth International Conference on Learning Representations*. 2.3.2

[110] Zachary Teed and Jia Deng. Tangent space backpropagation for 3d transformation groups. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 5.4.1, 6.3

[111] Emanuel Trabes and Mario A. Jordan. Self-tuning of a sunlight-deflickering filter for moving scenes underwater. In *2015 XVI Workshop on Information Processing and Control (RPIC)*, pages 1–6, 2015. doi: 10.1109/RPIC.2015.7497107. 3.2.1

[112] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T. Barron, and Pratul P. Srinivasan. Ref-NeRF: Structured view-dependent appearance for neural radiance fields. *CVPR*, 2022. 2.2.1

[113] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *CVPR*, 2024. 2.1.4

[114] Yuehao Wang, Chaoyi Wang, Bingchen Gong, and Tianfan Xue. Bilateral guided radiance field processing, 2024. 3.1

[115] Zhendong Wang, Xiaodong Cun, Jianmin Bao, Wengang Zhou, Jianzhuang Liu, and Houqiang Li. Uformer: A general u-shaped transformer for image restoration. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 17683–17693, 2022. 3.2.2

[116] Wikipedia contributors. Bathysphere — Wikipedia, the free encyclopedia, 2024. URL https://en.wikipedia.org/w/index.php?title=Bathysphere&oldid=1215499999. [Online; accessed 15-August-2024]. 1.1

[117] Wikipedia contributors. Autonomous underwater vehicle — Wikipedia, the free encyclopedia, 2024. URL https://en.wikipedia.org/w/index.php?title=Autonomous_underwater_vehicle&oldid=1225055301. [Online; accessed 15-August-2024]. 1.1

[118] Robert Woodham. Photometric method for determining surface orientation from multiple images. *Optical Engineering*, 19, 01 1992. doi: 10.1117/12.7972479. 6.1

[119] Jianfeng Xiang, Zelong Lv, Sicheng Xu, Yu Deng, Ruicheng Wang, Bowen Zhang, Dong Chen, Xin Tong, and Jiaolong Yang. Structured 3d latents for scalable and versatile 3d generation. *arXiv preprint arXiv:2412.01506*, 2024. (document), 7.2.2, 7.2

[120] Xiaohao Xu, Tianyi Zhang, Shibo Zhao, Xiang Li, Sibo Wang, Yongqi Chen, Ye Li, Bhiksha Raj, Matthew Johnson-Roberson, Sebastian Scherer, and Xiaonan Huang. Scalable benchmarking and robust learning for noise-free ego-motion and 3d reconstruction from noisy video. In *ICLR*, 2025. (document), 7.4.8, 8.2

[121] Disai Yang, Bingwei He, Mingzhu Zhu, and Jiantao Liu. An extrinsic calibration method with closed-form solution for underwater opti-acoustic imaging system. *IEEE Transactions on Instrumentation and Measurement*, 69(9):6828–6842, 2020. doi: 10.1109/TIM.2020.2976082. 5.2

[122] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. *Conference on Neural Information Processing Systems (NeurIPS)*, 2024. (document), 2.1.4, 7.2.3, 7.3, 7.3.1, 7.3.3, 7.7

[123] Wenqi Yang, Guanying Chen, Chaofeng Chen, Zhenfang Chen, and Kwan-Yee K. Wong. S^3-nerf: Neural reflectance field from shading and shadow under a single viewpoint. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2022. 2.2.1, 4.4.2

[124] Taoran Yi, Jiemin Fang, Junjie Wang, Guanjun Wu, Lingxi Xie, Xiaopeng Zhang, Wenyu

Liu, Qi Tian, and Xinggang Wang. Gaussiandreamer: Fast generation from text to 3d gaussians by bridging 2d and 3d diffusion models. In *CVPR*, 2024. 2.3.2

[125] Jason Y Zhang, Amy Lin, Moneish Kumar, Tzu-Hsuan Yang, Deva Ramanan, and Shubham Tulsiani. Cameras as rays: Pose estimation via ray diffusion. In *International Conference on Learning Representations (ICLR)*, 2024. 2.1.4

[126] Tianyi Zhang and Matthew Johnson-Roberson. Learning cross-scale visual representations for real-time image geo-localization. *IEEE Robotics and Automation Letters*, 7(2):5087–5094, 2022. doi: 10.1109/LRA.2022.3154035. 7.3.2, 7.4.2

[127] Tianyi Zhang and Matthew Johnson-Roberson. Beyond nerf underwater: Learning neural reflectance fields for true color correction of marine imagery. *IEEE Robotics and Automation Letters*, 8(10):6467–6474, 2023. doi: 10.1109/LRA.2023.3307287. (document), 2.2.2, 3.1, 7.4.8, 8.2

[128] Tianyi Zhang, Qilin Sun, and Matthew Johnson-Roberson. Learning neural reflectance fields for true color correction and novel-view synthesis of underwater robotic imagery. *IROS PIES Workshop*, 2023. (document), 8.2

[129] Tianyi Zhang, Kaining Huang, Weiming Zhi, and Matthew Johnson-Roberson. Darkgs: Learning neural illumination and 3d gaussians relighting for robotic exploration in the dark. *arXiv preprint arXiv:2403.10814*, 2024. (document), 3.1, 7.4.8, 8.2

[130] Tianyi Zhang, Weiming Zhi, Braden Meyers, Nelson Durrant, Kaining Huang, Joshua Mangelson, Corina Barbalata, and Matthew Johnson-Roberson. Recgs: Removing water caustic with recurrent gaussian splatting. *IEEE Robotics and Automation Letters*, 10(1):668–675, 2025. doi: 10.1109/LRA.2024.3511418. (document), 1.2, 7.4.8, 8.2

[131] Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000. doi: 10.1109/34.888718. 5.2

[132] Jiaxi Zheng, Guangmin Dai, Botao He, Zhaoyang Mu, Zhaochen Meng, Tianyi Zhang, Weiming Zhi, and Dixia Fan. Rs-modcubes: Self-reconfigurable, scalable, modular cubic robots for underwater operations. *IEEE Robotics and Automation Letters*, 10(4):3534–3541, 2025. doi: 10.1109/LRA.2025.3543139. (document), 8.2