Towards Practical Real-time Reflectance Estimation

Dissertation

zur
Erlangung des Doktorgrades (Dr. rer. nat.)
der
Mathematisch-Naturwissenschaftlichen Fakultät
der
Rheinischen Friedrich-Wilhelms-Universität Bonn

vorgelegt von

Lukas Bode

aus
Troisdorf

Bonn 2022

Angefertigt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät
der Rheinischen Friedrich-Wilhelms-Universität Bonn

1. Gutachter:	Prof. Dr. Reinhard Klein
2. Gutachter:	Prof. Dr. Michael Weinmann

Tag der Promotion:	01.02.2023
Erscheinungsjahr:	2023

Acknowledgements

First of all, I want to thank Reinhard Klein for supervising my Ph.D. studies and supporting me throughout my stay at the University of Bonn, starting with my Master's Thesis and ending with this dissertation. I also want to thank Michael Weinmann for giving lots of helpful advice along the way and reviewing this thesis, even after leaving the University of Bonn.

Further, thanks go to Patrick Stotko, who was a great officemate during my time in Bonn, supported my research projects through long and helpful discussions, as well as provided me with this awesome thesis template. I also want to thank the three previously mentioned people, as well as Sebastian Merzbach and Julian Kaltheuner, for co-authoring my publications. Without each and every one of you, my Ph.D. would not have been possible.

Additionally, I want to thank all my colleagues at the Visual Computing Institute at the University of Bonn. You made my studies a genuinely memorable experience.

I am also very thankful for my internships at Meta Reality Labs Research in 2020 and 2021. Special thanks go to Michael Goesele, Yujia Chen, Christophe Hery, and Carlos Aliaga for giving me these amazing opportunities and supervising me during the internships.

For proofreading this thesis, I want to thank Sebastian Kappes, Ralf Riesen, and Patrick Stotko.

Finally, I thank all of my friends and family for their tremendous support throughout my studies. It was a big help to know that I can always rely on you.

Abstract

Virtual experiences are becoming increasingly popular, primarily due to tremendous progress in *Virtual Reality* (VR) and *Augmented Reality* (AR) technologies improving immersion and making related devices more affordable. These immersive experiences frequently rely on the detailed and accurate reconstruction of real-world objects or people. While some applications can utilize assets captured offline in highly calibrated environments, others depend on real-time online scene reconstruction.

A complete reconstruction consists of information regarding geometry, illumination, and reflectance properties. Especially capturing the reflectance characteristics of real-world scenes is very challenging as it relies on the disentangling of intrinsic scene properties based on appearance samples. While a dense sampling and consecutive fitting of reflectance models may be feasible in an offline setting, this is not the case for applications depending on real-time reflectance estimation, as they usually impose additional constraints preventing a structured and controlled capturing process.

To this end, we identify two main challenges for the field of reflectance estimation in this thesis, which must be overcome to build practical real-time reflectance estimation pipelines: strong time constraints and sparsity of appearance samples. As part of the thesis, we present three previously published projects to address these: First, we propose a complete real-time reflectance estimation pipeline efficiently implemented on the GPU and leveraging deep learning techniques. Afterward, we explore denoising to restore reflectance estimates corrupted by noise-like artifacts due to the mentioned challenges. Finally, a novel lightweight edge and boundary detection approach is proposed to improve scene understanding and provide additional helpful information to reflectance estimation pipelines.

Contents

Part I

Introduction

Chapter 1

Introduction

With the recent popularity of *Metaverse* [Lee et al., 2021] as well as *Augmented Reality* (AR) and *Virtual Reality* (VR) related technology, the capturing of real-world scenes and creation of virtual duplicates is more important than ever. Being able to create truly immersive virtual experiences would have many benefits for our everyday life. The ecological footprint could be reduced due to less required traveling as many activities could be done using telepresence systems. People could interact socially over long distances without missing out on aspects of typical in-person social interaction, which would improve general mental well-being. Life would be more inclusive by allowing everyone to participate in society regardless of personal handicaps. Besides telepresence, there are numerous applications for immersive reconstructions in various fields, including entertainment, cultural heritage, and remote exploration of dynamic and dangerous environments.

While for many VR applications, it might be sufficient to bake the scene illumination into the appearance representation and only reproduce the captured scene in its entirety, for most AR applications, this is not enough. Many immersive AR applications require seamlessly integrating photorealistic virtual objects into the users' uncontrolled surroundings, which is generally only possible through the simulation of light transport in the scene.

A critical milestone towards scene reconstruction suitable for physically plausible light transport simulation is being able to truthfully reconstruct the appearance of a large variety of different scenes. What we perceive as appearance is a complex interplay of geometry, illumination, and the reflectance characteristics of materials. As most sensors capture only one or two of these components but rarely all three, we need algorithms to accurately reconstruct the unknown scene intrinsics based on incomplete data as well as possible.

Typically, priors are used to account for sparsely captured appearance data, which rely on strong assumptions on the captured scenes not universally holding for real-world data. Thus, these reconstruction algorithms often introduce artifacts to the scene, resulting in a loss of immersion and potentially virtual reality sickness. Moreover, due to the complexity of the problem and the large amount of data that must be processed to reconstruct a single scene, most appearance reconstruction algorithms are costly in terms of computational resources.

This thesis presents methods that tackle these open problems and, thereby, contribute towards the goal of practical real-time reflectance estimation from sparse data.

1.1 Challenges

The main challenges this thesis is concerned with can be summarized as follows:

Time Constraints Many applications, like the previously mentioned telepresence settings, require the latency of the capturing and reconstruction system to be in the order of seconds or even less. Higher latencies could, e.g., prevent suitable reactions to events in highly dynamic scenes or make social interaction challenging because of delayed feedback. Due to the sheer amount of data that has to be processed as part of these reconstruction systems, this problem is not trivial to solve if high-quality reconstructions are required. Especially the fitting of parameters for reflectance models often comes with an enormous computational burden depending on the underlying algorithm.

Sparse Data In most practical settings, dense sampling of the scene's appearance is not feasible due to constraints imposed by the targeted application. An exemplary setting is the exploration of hazardous environments with a remotely controlled robot as the appearance of the scene changes according to the view point, and the robot might not be able to maneuver the environment freely. Moreover, smooth surfaces can exhibit very sharp highlights. Capturing these highlights is crucial for successfully estimating high-quality reflectance properties so as to not estimate the respective surface to be overly rough due to the absence of sharp reflections. However, they are also easily missed as they might only be visible under particular view and illumination configurations. This problem is even more severe in dynamic scenes in which one of the inherent scene properties responsible for the scene's appearance, i.e., geometry, illumination, or reflectance characteristics, changes over time, requiring one to sample the temporal domain as well. In conjunction with time constraints that make processing large amounts of data challenging, the aspects mentioned earlier require specifically designing algorithms to estimate the scene's reflectance characteristics based on sparse data.

1.2 List of Publications and Contributions

I have contributed towards the goal of practical real-time reflectance estimation throughout multiple research projects. These projects are integral parts of this thesis. The publications, as well as short descriptions, are listed in the following:

- **Lukas Bode**, Sebastian Merzbach, Patrick Stotko, Michael Weinmann, and Reinhard Klein.
  "Real-time Multi-material Reflectance Reconstruction for Large-scale Scenes under Uncontrolled Illumination from RGB-D Image Sequences."
  International Conference on 3D Vision (3DV), 2019.
  DOI: `10.1109/3DV.2019.00083`
  Contribution: In this work, we propose a complete reflectance estimation pipeline for large-scale scenes. An efficient GPU implementation makes the algorithm capable of online estimation of reflectance parameters and real-time rerendering of the resulting virtual scene. The reflectance parameters are estimated from sparse data in a per-object manner by utilizing a deep neural network in conjunction with a geometry-based scene segmentation. Furthermore, a novel albedo refinement technique enables the reconstruction of objects with spatially varying reflectance properties.

- **Lukas Bode**, Sebastian Merzbach, Julian Kaltheuner, Michael Weinmann, and Reinhard Klein.
  "Locally-guided Neural Denoising."
  Graphics and Visual Computing (GVC), 2022.
  DOI: `10.1016/j.gvc.2022.200058`
  Contribution: Usually, time constraints can be satisfied by accepting lower-quality reconstructions, e.g., by using less data for the fitting of reflectance model parameters. While a low number of appearance samples may be sufficient to reconstruct rough materials accurately, the same number of samples may yield corrupted results for smooth materials due to a higher chance of sharp highlights not being represented. In this work, we propose restoring this partially corrupted reflectance data instead of modifying the capturing and fitting process. We adapt state-of-the-art denoising algorithms to utilize additional guidance information based on local noise-level estimates to remove artifacts but preserve fine details in initially clean regions.

- **Lukas Bode**, Michael Weinmann, and Reinhard Klein.
  "BoundED: Neural Boundary and Edge Detection in 3D Point Clouds via Local Neighborhood Statistics."
  arXiv preprint arXiv:2210.13305, submitted to ISPRS Journal of Photogrammetry and Remote Sensing (P&RS) (under review), 2022.
  DOI: `10.48550/arXiv.2210.13305`
  Contribution: The problem of sparse appearance samples for reflectance estimation can be tackled by assuming that multiple points of the scene exhibit similar reflectance behavior and, therefore, pooling the respective appearance samples to estimate a single set of reflectance model parameters. A common choice is to assume objects to consist of a homogeneous material. However, this assumption is quite limiting as many real-world objects are assembled from multiple materials. To alleviate the requirement of only capturing homogeneous objects, we can instead assume individual smooth surfaces of objects to consist of independent homogeneous materials. In this work, we propose a novel state-of-the-art algorithm for edge and boundary detection in point clouds, which could be used to find such a segmentation of the scene into individual smooth surfaces.

Besides the previously listed works, I also contributed to the following publication, which, however, is not part of this thesis:

- Julian Kaltheuner, **Lukas Bode**, and Reinhard Klein.
  "Capturing Anisotropic SVBRDFs."
  International Symposium on Vision, Modeling, and Visualization (VMV), 2021.
  DOI: `10.2312/vmv.20211372`

1.3 Thesis Outline

In the following, we provide an outline for the thesis:

First, important preliminary knowledge is summarized, and previous research related to practical real-time reflectance estimation is discussed in Chapter 2. Afterwards, a complete reflectance estimation pipeline [Bode et al., 2019] is described in Chapter 3. Next, the denoising of images partially corrupted with noise-like artifacts [Bode et al., 2022a] is explored in Chapter 4. Furthermore, a novel approach for lightweight boundary and edge detection in point cloud data [Bode et al., 2022b] is presented in Chapter 5. Finally, we conclude this thesis by summarizing our contributions, discussing limitations, and giving an outlook for the future of reflectance estimation pipelines in Chapter 6.

Chapter 2

Background and Related Work

This chapter aims to provide background information and list recent work in the research fields relevant to this thesis. First, the recent progress in general reflectance estimation solutions is discussed. We will cover approaches working on only a single or few images and ones working on whole image sequences. Then, the research background of algorithms for denoising and edge detection is given, as these are particularly important in the context of this thesis.

2.1 Reflectance Estimation

Appearance reconstruction is a versatile tool useful for a large variety of different applications. The general idea is to disentangle various appearance-related properties, i.e., geometry, illumination, and reflectance characteristics, of a scene based on suitable sensor data. While we can easily capture scene geometry using, e.g., Time-of-Flight (ToF) depth sensors and illumination can be captured using photo or video cameras, no such simple solution exists for capturing the reflectance of surfaces in the wild. Thus, the most common approach is to sample the appearance of surfaces using photo or video cameras and consecutively approximately invert the light transport in the scene.

Kajiya [1986] describes the light transport via the rendering equation

$$L_o(x, \omega_o) = L_e(x, \omega_o) + \int_\Omega f(\omega_i, x, \omega_o) L_i(x, \omega_i) \cos\theta_i \, d\omega_i, \tag{2.1}$$

where $L_o(x, \omega)$ and $L_i(x, \omega)$ are outgoing and incoming radiance in direction ω from and to surface point x respectively, $L_e(x, \omega)$ is the radiance emitted by the surface from surface point x in direction ω, $f(\omega_i, x, \omega_o)$ is the bidirectional reflectance distribution function (BRDF) describing the ratio of the radiance being reflected into direction ω_o to the irradiance from direction ω_i, θ_i is the angle between incoming light direction ω_i and the surface normal, and the integration domain Ω consists of all directions over a unit hemisphere in the direction of the surface normal. Intuitively, this equation describes the appearance of a surface point x from a given viewing direction ω_o, which depends on the surface normal, the incoming light from all possible directions, and the surface's reflectance characteristics described by the BRDF. Note that this formulation already contains several simplifications, including no transparent surfaces in the scene, no change of wavelength in the light due to fluorescence or phosphorescence, and no visible subsurface scattering effects. These assumptions are commonly employed to increase the feasibility of the inverse rendering problem.

Reflectance estimation pipelines try to recover the BRDF f for a captured scene. For this, early works often used multiple photographs captured under controlled environments. Goldman et al. [2009] propose a photometric stereo technique that reconstructs surface reflectance as a weighted sum of basis materials based on multiple photographs taken under various illumination configurations. Combining this with multi-view stereo techniques and an alternating geometry and spatially-varying bidirectional reflectance distribution function (SVBRDF) reconstruction, Ruiters and Klein [2009] developed a reconstruction method for near-planar material samples. Using this method, even global illumination effects like interreflections can be considered during reconstruction. Furthermore, materials exhibiting more complex reflectance characteristics can be captured by leveraging data-driven reflectance models [Ruiters et al., 2012].

However, despite simplifying the light transport simulation, the problem is still heavily underconstrained in more practical, less controlled settings as usually only a limited number of observations from very few directions under very few or even only a single light configuration are available to reconstruct all intrinsic scene properties. Capturing the illumination or geometry using additional hardware can reduce the ambiguity of the inverse rendering problem, but even in this case, the problem remains challenging due to global illumination effects like shadows and interreflections, as well as complex reflectance characteristics exhibited by some real-world materials.

Many applications impose additional requirements on the capturing process. Image editing, e.g., requires the algorithm to work on a single input image. In contrast, in telepresence applications, image sequences are usually available, allowing for the temporal fusion of the individual estimates. Hence, this section will touch on different research branches of appearance reconstruction.

Single- and Few-shot Decomposition Decomposing a given photo or rendering into its intrinsic components, i.e., geometry, illumination, and reflectance, has been an important research topic in computer vision for many decades. Already in the 1970s Barrow and Tenenbaum [1978] published fundamental work decomposing a given image into a product of a shading and a reflectance layer. Since then, it has been an active research area, and numerous improvements have been developed.

Based on an HDR image of a single object of known geometry consisting of a homogeneous material, Romeiro and Zickler [2010] can estimate the object's reflectance and illumination. For this, the reflectance is represented as a linear combination of materials from a database. A wavelet basis is used for illumination. In order to solve the inherent ambiguity of the problem, very strong statistical priors are employed by Barron and Malik [2014] to find the most probable decomposition of an image into shape, reflectance, and illumination.

For some applications, the decomposition into geometry and so-called reflectance maps, which combine the scene's illumination and reflectance characteristics, is already sufficient. An example can be transferring material properties between objects in an augmented reality context. *Deep Reflectance Maps* [Rematas et al., 2016] represent classical reflectance maps as a neural network. The inherent data interpolation capabilities of neural networks are leveraged to handle the sparse appearance data available in the single input image. Similarly, *Deep Appearance Maps* [Maximov et al., 2019] can be used instead if a more expressive representation is required for the desired application. Fitting such a neural appearance representation is computationally expensive. Good results have been achieved by training a separate neural network to predict *Deep Appearance Map* parameters from a given input image, also drastically reducing the time required to perform the image decomposition. This approach, however, introduces a strong dependence on the used dataset for training this *learning-to-learn* network, i.e., the network cannot predict *Deep Appearance Maps* for images showing objects which are not part of the training data. Assuming a homogeneous Phong BRDF [Phong, 1975], reflectance maps can also be decomposed into reflectance and illumination components consecutively [Georgoulis et al., 2016].

High-quality SVBRDFs can be captured by leveraging flash/no-flash image pairs [Aittala et al., 2015] by exploiting self-similarities. Despite yielding great results, this algorithm is restricted to flat material samples limiting its applicability for many use cases. The requirement of using flash/no-flash image pairs can be lifted in this setting by using self-augmented convolutional neural networks [Li et al., 2017].

Homogeneous BRDFs can also be estimated from more complex objects using a joint optimization for reflectance and illumination characteristics [Lombardi and Nishino, 2012a; Lombardi and Nishino, 2015]. Combining the color images with depth images [Lombardi and Nishino, 2016], these approaches can also be extended to a scene consisting of multiple objects. Furthermore, Meka et al. [2018] increased the performance and replaced used priors by utilizing carefully designed encoder-decoder neural networks trained on a large synthetic dataset in an end-to-end fashion.

The problem of only sparse appearance data being available in a single image can also be tackled by regularizing the reflectance estimation based on additional sensor data. To this end, noisy depth data from an RGB-D sensor has been used [Barron and Malik, 2013; Chen and Koltun, 2013; Hachama et al., 2015; Shi et al., 2015] as well as near-infrared images captured using respective filters [Cheng et al., 2019b] or hyperspectral images using a specialized camera [Zhang et al., 2022].

Finally, based on a single image and known geometry, Oechsle et al. [2019] have fomulated texture optimization as an adversarial problem yielding impressive results.

Additional related work can be found in recent state-of-the-art reports [Bonneel et al., 2017; Garces et al., 2022].

Offline Estimation from an Image Sequence In many cases, it is feasible to use an entire image sequence for appearance reconstruction instead of just a single or very few images.

While the resulting problem is still underconstrained, the additional data usually enables much more accurate results.

Early work in this research area relies on collections of images taken from the internet, with all images showing the same scene under possibly varying illumination. Haber et al. [2009] can estimate reflectance characteristics in this setting using a wavelet framework. The algorithm, however, requires accurate geometry of the scene as input. Similar work [Diaz and Sturm, 2013] is capable of simultaneously estimating the radiometric camera response curve.

In the particular case of reconstructing the appearance of a flat surface, Albert et al. [2018] propose to use a mobile phone video as input. By placing additional auxiliary markers in the scene, high-quality SVBRDF estimates can be achieved.

Several works have been published to recover the reflectance characteristics of a single object from video input. Utilizing given accurate geometry of the object as additional input, spatially varying diffuse albedo as well as per-cluster homogeneous Phong BRDF [Phong, 1975] parameters can be reconstructed based on statistical priors [Palma et al., 2012]. Under the additional requirement of the video showing a rotating object from a fixed view point, even more expressive spatially-varying microfacet BRDF parameters can be estimated [Dong et al., 2014].

Based on RGB-D data captured with commodity sensors, a joint estimation of geometry and reflectance [Wu et al., 2015] can be performed to alliviate the need for a given accurate geometry while at the same time leveraging the low noise-level in RGB images to improve the reconstructed geometry. Learning-based techniques are also used to estimate homogeneous Ward BRDF [Ward, 1992] parameters based on RGB-D data unter uncontrolled but static illumination [Kim et al., 2017]. The network for this algorithm is trained on synthetic data only and therefore avoids the need for an expensive collection of real-world training data. More complex scenes consisting of multiple objects can be reconstructed based on a segmentation of the scene [Richter-Trummer et al., 2016] or based on an initial signed distance field (SDF) geometry reconstruction with a subsequent coarse-to-fine refinement of estimated geometry and albedo estimates [Maier et al., 2017].

Another branch of research focuses on capturing color textures from RGB-D data. Applying a global optimization scheme to align the observations from various view points yields high-quality textures capturing even fine details [Zhou and Koltun, 2014; Bi et al., 2017; Fu et al., 2021]. Furthermore, color textures can also be obtained through adversarial machine-learning techniques [Huang et al., 2020]. While such a texture captures fine details of the scene very well, these approaches are not applicable to tasks requiring relighting as no physically-based reflectance is captured.

Differentiable Rendering Recently, differentiable rendering became a prevalent tool for appearance reconstruction tasks. More and more differentiable rendering frameworks are published, ranging from general purpose differentiable path-tracers [Li et al., 2018; Nimier-David et al., 2019] over more light-weight solutions [Zhang et al., 2020; Lassner and

Zollhofer, 2021] to differentiable rasterization-based renderers [Laine et al., 2020]. With the help of these frameworks, a gradient with respect to scene parameters like illumination, geometry, or reflectance can be calculated for a rendered image, which allows for stochastic gradient descent-based optimization to find parameters matching one or more given input images.

A rendering-aware network layer is used by Liu et al. [2017] to allow for optimization-based material editing of an object in a given image. Azinovic et al. [2019] use a differentiable rendering formulation to estimate a scene's material properties and illumination based on given RGB images and the scene's geometry. To make this optimization problem practical, they fit one set of reflectance parameters per object based on a scene segmentation. Moreover, by leveraging a cascaded network with a differentiable rendering layer, reflectance parameters for indoor scenes can be estimated with high quality from a single image [Li et al., 2020], allowing for realistic insertion of virtual objects into the scene, including global illumination effects like shadows or interreflections. Similarly, Dai et al. [2021] utilize differentiable rendering to improve the alignment of textures in an RGB-D data-based reconstruction. Besides mesh and SDF-based scene representation, differentiable rendering is also used in conjunction with surface splatting [Müller et al., 2022a].

Another way to approach the problem is by training a neural network to perform the image formation process, as neural networks are inherently differentiable. Nguyen-Phuoc et al. [2018] demonstrate the success of this technique for rendering and inverse-rendering tasks. While this uses a classical scene representation, Thies et al. [2019] use a proxy geometry and a neural texture to reconstruct scenes with complex appearance. This is possible by training both neural texture and rendering network in an end-to-end fashion.

For novel view synthesis tasks, *Neural Radiance Fields* (NeRFs) [Mildenhall et al., 2021] have tremendously improved the state of the art using an MLP to represent the scene as a combination of opacity and color values trained using a differentiable volumetric rendering loss function. Building on the initial work, many improvements have been proposed, including the usage of a spatial hashing scheme to accelerate training and inference performance [Müller et al., 2022b], tiling of individual NeRFs to enable city-scale reconstruction [Tancik et al., 2022], handling of dynamic scenes through learning an additional mapping of spatial positions to a canonical NeRF for each timestep [Pumarola et al., 2021], leveraging multi-scale representations to reduce aliasing artifacts [Barron et al., 2021], utilizing a distortion-based regularizer to handle unbounded scenes [Barron et al., 2022] and many more.

Recently, a new differentiable rendering technique [Vicini et al., 2021] was published, reducing the memory footprint and improving performance. Despite these recent improvements and results of impressive quality, differentiable rendering approaches are too slow for most time-critical applications.

A complete overview of recent publications in the field of differentiable and neural rendering is given in the recent survey by Tewari et al. [2022].

Online Estimation from an Image Sequence The KinectFusion [Izadi et al., 2011; Newcombe et al., 2011] algorithm was the first to enable online 3D scene reconstruction from RGB-D data obtained with consumer-grade depth sensors. Through an efficient registration and consecutive integration of new depth images on the GPU, room-scale scenes can be scanned with real-time visual feedback. In the following years, many reflectance estimation algorithms included a geometry reconstruction relying on this algorithm or one of its many successors improved, e.g., regarding the maximum size of the scene or globally consistent camera pose estimation [Nießner et al., 2013; Kähler et al., 2015a; Kähler et al., 2016; Dai et al., 2017].

Modern voxel-based online geometry reconstruction pipelines working on RGB-D data consist of the following basic steps:

1. **Camera Pose Estimation and Reprojection**: First, the camera pose of the new frame is estimated to reproject the depth images into the global 3D coordinate system. The pose estimation is usually conducted by finding a transformation between the depth data of the current frame and the reconstructed scene model up to the last frame.

2. **SDF Update**: The scene geometry is internally represented as a signed distance field (SDF) stored in a voxel grid, i.e., each voxel holds the distance to the nearest surface. This representation allows for an efficient update according to the data of the new frame by using a weighted averaging scheme.

3. **Surface Extraction**: After the new depth data has been integrated into the global scene geometry representation, an explicit surface can be generated from the SDF using, e.g., the ray marching [Hart et al., 1989] or marching cubes [Lorensen and Cline, 1987] algorithms.

Commodity RGB-D sensors often use automatic exposure adjustment to yield reasonable images in various illumination settings. This can be problematic for interpreting measured color values over multiple frames and the respective data fusion throughout the image sequence. The *HDRFusion* [Li et al., 2016] algorithm tackles this problem using an intermediate exposure estimation step in the reconstruction pipeline, which allows them to reconstruct HDR color maps for the scene.

By using an additional camera with a fish-eye lens to capture the environment, Knecht et al. [2012] can estimate Phong BRDF [Phong, 1975] parameters for the scene. While building on a geometry reconstruction framework with real-time capabilities, they do not reach real-time framerates for the reflectance estimation.

Kerl et al. [2014] were the first to utilize the infrared (IR) light emitted by ToF sensors like the Microsoft Kinect. Since this IR light has a much higher intensity than natural IR light in typical scenes, the captured IR images can be interpreted as illuminated by a single colocated point light source. This insight can be used to employ additional regularization to the estimation of reflectance in the spectrum of visible light. The *AppFusion* [Wu and Zhou, 2015] algorithm combined an IR-based clustering with capturing the environment by means of a mirror sphere in the scene in a multi-stage approach. Hereby, it can estimate high-quality Ward BRDF [Ward, 1992] parameters for a scanned object. Moreover, Stotko et al. [2019] utilize an additional object-based scene segmentation to further improve the coupling between RGB and IR images to estimate reflectance parameters for the scene.

With a focus on the application of live video editing, Meka et al. [2016] can decompose a scene into reflectance and shading layers in real time based on RGB video input. This approach yields impressive results for scenes in which the underlying assumption of purely lambertian reflections holds. These real-time video editing capabilities can be extended further to enable live user-guided editing [Meka et al., 2017] by utilizing RGB-D input data and storing user-specified constraints on a proxy geometry.

Increasingly more accurate geometry reconstructions enabled Whelan et al. [2016] to approximate the scene's illumination through a set of point light sources distributed over the scene through a voting-based approach. For this, specular highlights in the color images are detected, and corresponding rays are traced through the scene based on the reconstructed geometry. Consecutively, point light sources are placed in voxels that were hit by the highest number of rays. Omitting the need for depth input, Meka et al. [2021] employ clustering and an alternating optimization scheme of illumination and per-cluster base colors to decompose the video frames into a single direct and multiple indirect illumination layers.

Similar to the offline reconstruction setting, the highest amount of detail can be captured by means of fusing RGB images into textures for the geometry reconstruction. Lee et al. [2020] propose to store texture tiles in a voxel representation of the scene together with a spatially-varying perspective mapping. In contrast, the *TextureMe* [Kim et al., 2022] algorithm fuses color images into a global texture atlas.

As part of this thesis, we present a pipeline for voxel-based real-time reflectance estimation [Bode et al., 2019]. Through an efficient GPU implementation, spatially-varying diffuse albedo and per-object specular Ward BRDF [Ward, 1992] parameters can be reconstructed online.

2.2 Denoising

One of the main challenges in real-time reflectance estimation is the sparsity of available appearance samples, frequently leading to reconstructions being corrupted with noise-like artifacts. A similar challenge exists in the field of real-time path tracing, where integrals in the image formation process are approximately solved using Monte Carlo methods. Due to limited computational budgets, this approximation is usually made based on only very few random samples, leading to high variance in the solutions and, hence, noisy renderings. Very impressive results have recently been achieved by applying learning-based denoising algorithms to the resulting noisy renderings. The frequently used OptiX ray tracing engine [Parker et al., 2010] utilizes a recurrent autoencoder network [Chaitanya et al., 2017] for this purpose. In addition to the noisy color image, the authors use the albedo, normal, and surface roughness information to generate the denoised output image. Another example uses a so-called *ImportanceNet* to generate filtering kernels from similar input and recombines the filtered noisy irradiance images to generate a clean image [Fan et al., 2021].

Inspired by the success of such techniques, we explored denoising for artifact removal in reflectance estimation pipelines. Opposed to the path tracing setting, we do not have highly-accurate intrinsic information about the scene available in our case. Thus, we focus the discussion on previous works regarding general image denoising.

Still popular for denoising tasks is the application of total variation-based methods [Chambolle, 2004; Osher et al., 2005]. While computationally cheap, these algorithms tend to introduce blurring artifacts and lose fine image details. Similar artifacts can be observed using the non-local means approach by Buades et al. [2005] for images without self-similarities. A dictionary of image patches learned from either a separate dataset or the noisy input image itself can also be used as prior for denoising [Elad and Aharon, 2006]. Depending on the dictionary's contents, this algorithm can yield excellent results. However, the processing time is significantly higher than for the previously mentioned total variation-based algorithms. As an alternative to the learned dictionary, a learned gaussian mixture model prior is used by Zoran and Weiss [2011]. Finally, a gathering of similar image regions and consecutive filtering was proposed by Dabov et al. [2007] and demonstrated to work well on natural images.

Consecutive research [Foi et al., 2006; Dabov et al., 2007a; Miyata, 2015] assessed the importance of the color space to conduct the denoising in. Results suggest that jointly handling RGB color channels is beneficial over separately handling them or transforming the input data to a luminance-chrominance space.

In recent years, impressive results have been achieved by utilizing neural networks for denoising. Zhang et al. [2017] propose a novel convolutional neural network leveraging dilated convolution layers to enlarge the receptive field compared to ordinary convolution layers. The network is trained in a supervised manner on a large dataset. The denoising performance depends, therefore, on how well the training data matches the input data in terms of noise characteristics and content. Other supervised methods use, e.g., network ensambles [Yang et al., 2020] or a complex-valued convolutional neural network [Quan et al., 2021].

To lift the requirement of large datasets containing clean/noisy-image pairs exhibiting the targeted noise and content characteristics being available, which can be quite limiting in practice, Pang et al. [2021] developed a novel mathematical framework to train their network on noisy images only by corrupting them with additional additive noise. Thus, no clean images corresponding to the noisy training images are needed for their algorithm.

Going one step further, the *Deep Image Prior* (DIP) algorithm [Ulyanov et al., 2018] is a neural network-based method needing no training data at all as it is self-supervised. The authors found that convolutional neural networks have inherent regularization capabilities. Thus, by overfitting a network to map random noise to a single noisy input image, natural image content is learned before the network learns to represent the noise. A clean image can be extracted by interrupting the overfitting process before noise can be learned. As determining the correct number of training iterations after which to abort the training process can be difficult in practice, several improvements [Cheng et al., 2019a; Heckel and Hand, 2019; Kattamis et al., 2019] were proposed to stabilize the convergence behavior of the algorithm.

Furthermore, neural architecture search techniques [Chen et al., 2020; Ho et al., 2021] were combined with the *DIP* method to finetune the underlying architecture automatically to the specific denoising problem at hand. Other self-supervised denoising methods applicable in this setting include *Self2Self* [Quan et al., 2020] and *CVF-SID* [Neshatavar et al., 2022].

Most state-of-the-art denoising algorithms assume noise to have the same characteristics over the whole input image. In the context of reflectance estimation, however, noise-like artifacts can occur spatially concentrated, as, e.g., sharp highlights on smooth surfaces can easily be missed, while reconstructing the reflectance of perfectly diffuse surfaces is less prone to such errors. Applying off-the-shelf denoising solutions in this context usually leads to losing fine details in initially clean regions. At the same time, strong noise-like artifacts may not even be eradicated, depending on the algorithm. As part of this thesis, we present our novel solution for denoising of partially noisy images [Bode et al., 2022a].

2.3 Edge and Boundary Detection

A more direct option to handle the sparsity of appearance samples in reflectance estimation pipelines is the utilization of high-level scene understanding, e.g., in the form of a scene segmentation concerning objects or materials. The effectiveness of this technique has been demonstrated numerous times [Richter-Trummer et al., 2016; Azinovic et al., 2019; Bode et al., 2019; Stotko et al., 2019b] in recent years. While such a per-object segmentation is simple if the scene geometry is given as a mesh since topological information can be used to identify connected components, it is much more challenging in the case of popular point-based geometry representations using, e.g., disks [Whelan et al., 2016] or spheres [Lassner and Zollhofer, 2021] as underlying primitives. Besides reconstruction techniques using color and ToF sensors, point clouds are frequently used to represent laser-scanned geometry and gain increasing support in commercially used real-time rendering engines like *Unreal Engine* [Epic Games, 2022]. To foster the use of segmentation in point-based reflectance reconstruction methods and since it is crucial for high-level scene understanding, we explore methods for accurately identifying edges and boundaries in point clouds in the scope of this thesis.

Many algorithms for edge detection in point clouds rely on defining a continuous surface based on the points. The point set surfaces (PSS) algorithm [Alexa et al., 2001] does this based on the moving least squares (MLS) technique. Given a set of points $\mathcal{P} = \{p_i\}$ for $i = 1, \dots, n$, the surface is locally regressed for each point $r \in \mathcal{P}$. First, a local reference coordinate system for r is defined based on a least-squares fitted tangent plane. Afterward, a polynomial approximation $p(x, y)$ of the surface parameterized over the tangent plane domain is calculated by minimizing

$$\sum_{i=1}^{n} (p(x_i, y_i) - f_i)^2 \, \theta(||q - p_i||), \tag{2.2}$$

where x_i and y_i are the coordinates of the projection of p_i parameterized over the tangent plane, f_i is the distance of point p_i to the local tangent plane, q is the projection of r onto the tangent plane, and θ is a smooth weighting function, which reduces the influence of samples further away from q. This approximation has shown to be continuously differentiable if a suitable weighting function θ is chosen. Fleishman et al. [2005] propose to extract edges from point cloud data by intersecting multiple MLS-based smooth surfaces per local neighborhood. The PSS technique was extended by Guennebaud and Gross [2007] by projecting the points on fitted algebraic spheres instead of tangent planes improving the capabilities of the algorithm to capture sharp geometric features while still being able to naturally represent planar geometry.

Other works have addressed the problem of identifying sharp geometric features by analyzing the covariance matrices of local point neighborhoods. While Gumhold et al. [2001] initially proposed to conduct this analysis on a single scale, better results were achieved by conducting this analysis on multiple scales in parallel [Pauly et al., 2003]. This multi-scale technique was extended even further by Mellado et al. [2012] to an MLS-based surface representation continuously differentiable with respect to spatial position as well as the scale of the weighting function θ.

Weber et al. [2010] propose to analyze local neighborhoods concerning the normal distribution of all possible triangulations. Most normals point in a single direction for planar surfaces, while two or more distinct clusters exist for neighborhoods of points on sharp edges or corners. If the point cloud is not equipped with normals already, they can be estimated robustly concerning noise and outliers using e.g. a kernel density estimation-based technique [Li et al., 2010]. In the case of laser scans captured from a single position, Che and Olsen [2018] suggest that it is sufficient to consider only a single triangulation connecting the currently analyzed points with its direct neighbors in the scanning grid. Furthermore, the triangulation can be omitted entirely if a plane is obtained from the local neighborhood via *RANSAC* with a consecutive analysis of the neighborhood's points with respect to this plane [Lin et al., 2015].

Other methods for edge detection include utilization of alpha-shapes [Edelsbrunner and Mücke, 1994], covariance analysis of the point cloud's voronoi cells [Mérigot et al., 2011], spline fitting [Daniels Ii et al., 2008], mean-shift clustering [Ahmed et al., 2018], and automatic detection of planes [Mitropoulou and Georgopoulos, 2019].

Besides edge detection, boundary detection was addressed by Bendels et al. [2006]. The authors calculate the probability of a point being part of a boundary by evaluating several boundary criteria in local neighborhoods and combining the individual boundary probabilities using a weighted sum. Furthermore, Nguyen et al. [2015] suggest extracting an exterior boundary of the point cloud by analyzing neighborhood characteristics and afterward using a growth function to identify interior boundaries in the data. In contrast, Mineo et al. [2019] propose to estimate the point resolution in local neighborhoods of the point cloud and additionally calculate the mean radius of circles defined by the currently considered point and any two points for its neighborhood. Comparing these two quantities allows them to classify points as being part of a boundary or not.

Inspired by the success of neural network-based methods in many research fields, similar techniques were also applied to the edge and boundary detection problem. Sharpness fields can be extracted leveraging a local radial grid parameterization and a convolutional neural network (CNN) trained in a supervised manner [Raina et al., 2018; Raina et al., 2019]. High values in these sharpness fields correspond to sharp geometric features in the point cloud. *EC-Net* [Yu et al., 2018] identifies edges by encoding local neighborhoods with an underlying *PointNet++* [Qi et al., 2017b] and utilizing an edge-aware loss. Similarly, *PIE-NET* [Wang et al., 2020] uses two *PointNet++*-like networks and a consecutive non-maximal suppression technique to find the edge and corner points in point clouds. *PCPNet* [Guerrero et al., 2018] even uses a network inspired by *PointNet* [Qi et al., 2017a] in conjunction with multi-scale patches as input for estimating normals or curvatures as well as for point classification tasks. Furthermore, learning-based boundary detection was conducted, e.g., using graph convolutional networks [Loizou et al., 2020] or by feeding the whole point cloud into a deep neural network invariant to permutations of input points, that outputs a classification label per point.

While yielding impressive results, the computational burden of these approaches is immense. As these are, therefore, not feasible for usage in time-critical applications, PCEDNet [Himeur et al., 2021] addresses this problem by being designed to be lightweight and very fast. The authors achieve this by first computing growing least squares (GLS) [Mellado et al., 2012] features for all points on multiple scales, as these describe the point's local neighborhood very efficiently. Due to this efficient encoding of neighborhood information, a very small MLP is already sufficient to classify points as edge or non-edge based on these features.

As part of this thesis, we present a novel set of features [Bode et al., 2022b], which can be used similarly to the GLS features used by PCEDNet as input for a compact MLP for point cloud edge and boundary detection. We are able to outperform the state of the art in terms of processing time as well as classification performance.

Part II

Publications

Chapter 3

Real-time Multi-material Reflectance Reconstruction for Large-scale Scenes under Uncontrolled Illumination from RGB-D Image Sequences

In this chapter, we discuss the contributions and results developed in the following peer-reviewed publication:

> Lukas Bode, Sebastian Merzbach, Patrick Stotko, Michael Weinmann, and Reinhard Klein.
> "Real-time Multi-material Reflectance Reconstruction for Large-scale Scenes under Uncontrolled Illumination from RGB-D Image Sequences."
> *International Conference on 3D Vision (3DV)*, 2019.
> DOI: `10.1109/3DV.2019.00083`

3.1 Summary of the Publication

Capturing real-world scenes and creating realistic virtual duplicates has many applications in fields like telepresence, cultural heritage, and entertainment. Reconstructing a scene's appearance accurately is essential for many kinds of immersive virtual experiences. While geometry can be measured directly via depth sensors, reflectance can only be measured indirectly as appearance, which is the complex interplay between geometry, illumination, and material-specific reflectance characteristics. Due to the usually sparse nature of available appearance samples, this disentangling is an underconstrained problem that most classical fitting algorithms fail to solve well. In this work, we apply ideas from deep learning research to the problem of reflectance estimation in a real-time voxel-based 3D reconstruction pipeline.

In our pipeline, RGB-D frames are directly processed online when made available by a commodity RGB-D sensor like the Microsoft Kinect or modern smartphones equipped with a depth sensor. The goal is to produce a flexible representation of the scene, enabling efficient, high-quality rendering of the virtual scene from arbitrary views while continuously being updated as new sensor data becomes available.

To achieve this, the first step of our approach is to capture the scene's illumination via a separate image sequence in which the sensor is directly pointed toward the dominant light sources. Overexposed pixels in the RGB images are identified and back-projected into 3D space based on the captured depth values and estimated camera poses. In these bright areas, point light sources are distributed to approximate the scene's overall illumination. After this preliminary lighting estimation, a second image sequence can be recorded and processed online. The processing of the respective RGB-D frames consists of multiple steps: A voxel-based geometry reconstruction using the implementation of Stotko et al. [2019] is performed first. During this geometry reconstruction, we also store so-called observations per voxel, i.e., a pair of color and direction from which the color was observed for each frame. As it is not feasible to store every single observation of the whole image sequence due to memory limitations, we store only 30 observations per voxel on a separate voxel grid with lower resolution compared to the voxel grid used for the geometry reconstruction. Based on the reconstructed geometry, the scene is segmented into individual objects using the segmentation algorithm by Tateno et al. [2016]. Those objects are assumed to have a homogeneous specular reflectance behavior, that is estimated based on the aggregated observations by utilizing a siamese convolutional neural network (CNN). Based on the gathered observations as well as the estimated illumination, geometry, and specular reflectance parameters, a spatially-varying diffuse albedo is calculated to enable the reconstruction of fine details. To ensure temporal consistency and utilize all available information to the fullest extent, the parameters estimated per frame are fused over the whole image sequence up to the currently processed frame. All of these steps are implemented efficiently on the GPU. Due to the inherently parallel nature of most steps, framerates of around 30 Hz were achieved in real-world experiments on consumer-grade GPUs.

We evaluated this novel material estimation pipeline using synthetic and real-world data captured using a Microsoft Kinect v2 sensor. The evaluation shows that our pipeline can separate diffuse and specular appearance components well in most cases. Despite some instability in the estimates during the first few frames, the results stabilize quickly once enough appearance samples are gathered for the reflectance estimation CNN to yield meaningful results. Furthermore, our experiments show that the spatially-varying diffuse albedo refinement step is crucial for high-quality rerenderings of the scene, as many real-world objects are textured to varying degrees.

3.2 Author Contributions to the Publication

This work is building on my master's thesis [Bode, 2018], which implements a pipeline for the estimation of homogeneous per-object materials partially running on the GPU. For this publication, I introduced an additional illumination estimation step to the pipeline. Utilizing this information, I also developed the spatially varying albedo refinement. Additionally, I ported parts of the pipeline previously executed on the CPU to the GPU to increase the run-time performance and free up CPU resources simultaneously. Finally, I also captured further image sequences and conducted respective evaluation experiments to ensure a meaningful assessment of the algorithm.

Chapter 4

Locally-guided Neural Denoising

In this chapter, we discuss the contributions and results developed in the following peer-reviewed publication:

> Lukas Bode, Sebastian Merzbach, Julian Kaltheuner, Michael Weinmann, and Reinhard Klein.
> "Locally-guided Neural Denoising."
> *Graphics and Visual Computing (GVC)*, 2022.
> DOI: `10.1016/j.gvc.2022.200058`

4.1 Summary of the Publication

Measured data is often corrupted with noise-like artifacts. These artifacts can be introduced by the sensor itself due to physical or economic constraints or during the processing of the sensor data due to, e.g., a limited computing budget. While measures can be taken to reduce the amount of noise introduced during the capturing or processing of data, usually, it cannot be avoided altogether. Depending on the application of the captured data, this can be a problem, as, e.g., noise-like artifacts can break immersion in AR or VR settings. The corrupted data might also pose a challenge to further processing as many algorithms are designed for and tested on clean data only. Without directly interfering with the capturing process, image restoration algorithms can be used to mitigate this problem by enforcing additional priors in the data.

In the context of practical reflectance estimation, fitting reflectance model parameters to measured appearance samples may result in noise-like artifacts concentrated on very smooth surfaces due to an insufficient number of samples being available. Off-the-shelf denoising algorithms, however, either fail to remove the artifacts or blur data not exhibiting any noise resulting in a loss of fine details in those parts.

In this work, we propose extracting noise-level estimates from the input data and using them to guide state-of-the-art neural denoising algorithms. This additional information allows the denoising algorithm to restore corrupted data areas while ensuring that initially clean data is preserved.

The first step of our method is to estimate the spatially-varying noise level for the input data. Although we assume the corrupted data to be given as a 2D image, the general ideas can easily be translated to 3D data. To estimate the noise level, we propose two different solutions: The first option is to use a per-pixel variance calculated on a small neighborhood of pixels. While yielding good results in most cases, this method fails, e.g., for smooth color gradients or discontinuities in the image, which would erroneously yield high noise-level estimates. These problems are resolved in the second noise-level estimation method: First, the image data is lifted into a 3D space based on the pixel values and coordinates. Afterward, for each point, a plane is fitted to its local neighborhood in a least-squares manner, followed by partitioning the data points of the neighborhood into points on one side of the fitted plane and points on the other, respectively. Conducting a covariance analysis of the disjoint subsets of points of this local neighborhood finally enables us to calculate a per-point noise level that is robust against the smooth gradients and discontinuities mentioned above.

The noise level estimate calculated using either of these methods is subsequently used to inject guidance information into state-of-the-art neural denoising algorithms with only minor modifications. Exemplary, we demonstrate this on two different approaches: *Deep Image Prior* (DIP) [Ulyanov et al., 2018] and *Self2Self* (S2S) [Quan et al., 2020]. In DIP, which denoises data using the inherent regularization capabilities of convolutional neural networks, the guidance information is injected via a modified loss function. In case of S2S, which uses probabilistic input masking and dropout to denoise images, we instead apply the guidance map guidance map to modify the probabilities of the input masking and the dropout layers on a per-pixel basis. Since the guidance information can effectively be utilized in these two popular state of the art denoising algorithms, we expect it can also be integrated easily into future denoising approaches.

The noise-level estimates are assessed qualitatively on fitted SVBRDF textures which exhibit strong spatially concentrated noise-like artifacts. We show that in most cases, corrupted regions yield high noise-level values, while they are usually low for clean regions. Consequently, when using these noise levels as guidance for denoising, the artifacts are significantly reduced. A quantitative comparison to other denoising algorithms shows improvements over the current state-of-the-art. Similar results are also achieved on real-world photos corrupted with additive gaussian white noise.

4.2 Author Contributions to the Publication

In the scope of this work, I implemented both noise-level estimation methods efficiently on the GPU. Furthermore, I implemented the modified DIP algorithm from scratch and modified an open-source S2S implementation to leverage the additional guidance information as described above. I conducted all experiments used to evaluate the proposed methods against the state of the art. The code to read and render the SVBRDF texture data was supplied by Julian Kaltheuner and Sebastian Merzbach.

Chapter 5

BoundED: Neural Boundary and Edge Detection in 3D Point Clouds via Local Neighborhood Statistics

In this chapter, we discuss the contributions and results developed in the following publication, which already appeared as a preprint and is currently under review:

> Lukas Bode, Michael Weinmann, and Reinhard Klein.
> "BoundED: Neural Boundary and Edge Detection in 3D Point Clouds via Local Neighborhood Statistics."
> *arXiv preprint arXiv:2210.13305, submitted to ISPRS Journal of Photogrammetry and Remote Sensing (P&RS) (under review)*, 2022.
> DOI: `10.48550/arXiv.2210.13305`

In the following, we include a verbatim copy of the content of this work subject to some minor editorial changes.

Author Contributions to the Publication In this work, together with my co-authors, I developed the idea of using various statistical features as input for the classification network. I implemented the whole pipeline consisting of feature extraction and training as well as inference of the classification network on various datasets. I conducted all experiments using the previously mentioned implementation of our novel algorithm and partially adapted publicly available implementations of related work.

5.1 Abstract

Extracting high-level structural information from 3D point clouds is challenging but essential for tasks like urban planning or autonomous driving requiring an advanced understanding of the scene at hand. Existing approaches are still not able to produce high-quality results

consistently while being fast enough to be deployed in scenarios requiring interactivity. We propose to utilize a novel set of features describing the local neighborhood on a per-point basis via first and second order statistics as input for a simple and compact classification network to distinguish between non-edge, sharp-edge, and boundary points in the given data. Leveraging this feature embedding enables our algorithm to outperform the state-of-the-art techniques in terms of quality and processing time.

5.2 Introduction

3D point cloud data obtained from terrestrial or airborne laser scanning as well as depth sensors and image-based structure-from-motion have become the prerequisite for numerous applications including geographic information systems, urban planning, indoor modeling for the built environment, autonomous driving, and navigation systems. However, the sampling of scenes with arbitrary complexity in terms of unstructured data complicates the further processing of the data as e.g. required when extracting characteristic features for navigation or scene interpretation according to object instances and materials. Edges represent characteristic features that often occur at object borders as well as on surfaces (in the form of ridges or engravings) and linear scene structures like scaffolds and, hence, provide essential information regarding the underlying geometric structures. However, automatic edge detection in 3D point cloud data remains a challenging task. Whereas physical edges may not appear as sharp due to damage or cleaning (e.g. stone or plastered buildings, progressively smoothed edges, polished mechanical parts, etc.), there are also limitations inherent to the scanning approaches, especially due to the typically uneven, noisy sampling of the scene, that may result in a slight rounding effect of edges in the reconstruction. Furthermore, the sharpness, smoothness or roundness of edges also depends on the observation scale. Therefore, there might be some ambiguity in defining edges, that may require involving further context information. In addition, with point clouds typically consisting of tens or hundreds of millions of points, efficient operators are required.

Advances in machine learning and the rapidly growing availability of 3D data have led to several supervised learning approaches for concept classification. Respective approaches include the classification of structures according to semantic categories such as facades, roofs, different forms of vegetation or pole/trunk structures using pointwise hand-crafted geometric descriptors on a single *optimal* scale [Demantké et al., 2011; Weinmann et al., 2015a; Weinmann et al., 2015c; Hackel et al., 2016b] or multiple scales [Brodu and Lague, 2012; Blomley and Weinmann, 2017], additionally leveraging contextual information [Niemeyer et al., 2014; Weinmann et al., 2015b; Landrieu et al., 2017; Steinsiek et al., 2017], as well as deep-learning strategies [Huang and You, 2016; Boulch et al., 2017; Hackel et al., 2017; Lawin et al., 2017; Qi et al., 2017b; Tchapmi et al., 2017; Landrieu and Simonovsky, 2018; Thomas et al., 2019; Guo et al., 2020; Xie et al., 2020; Li et al., 2022; Mao et al., 2022]. Furthermore, a few works also focused on the individual classification of points according to being or not being on edges based on multi-scale features and a random-forest-based classification [Hackel et al., 2016a], multi-scale features and a dedicated neural network based edge detection classifier [Himeur et al., 2021], neural-network-based pointwise distance

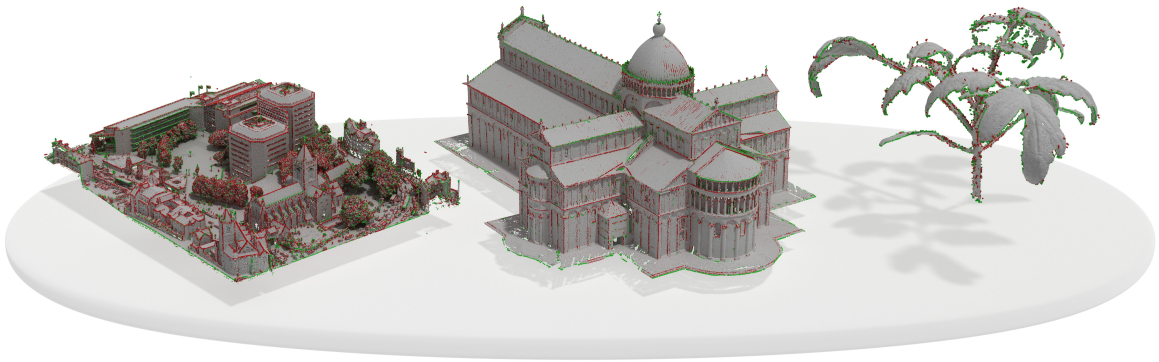

Figure 5.1: Our BoundED approach extracts sharp edges and boundaries from 3D point cloud data purely based on positional data. Points classified as sharp-edge are highlighted in red while boundary points are highlighted in green.

estimation to the next sharp geometric feature [Matveev et al., 2022], binary-pattern-based filtering on local topology graphs [Guo et al., 2022], neural-network-based edge-aware point set consolidation leveraging an edge-aware loss [Yu et al., 2018], training two networks based on PointNet++ [Qi et al., 2017b] to classify points into corners and edges and subsequently applying non-maximal suppression and inferring feature curves [Wang et al., 2020], the learning of multi-scale local shape properties (e.g., normal and curvature) [Guerrero et al., 2018], and the computation of a scalar *sharpness* field defined on the underlying Moving Least-Squares surface of the point cloud whose local maxima correspond to sharp edges [Raina et al., 2018; Raina et al., 2019]. However, extracting high-quality edge and boundary data from a large variety of different 3D point clouds fast enough to eventually be suitable for usage in embedded systems or real-time settings remains an open problem.

In this paper, inspired by the *maximum mean discrepancy* (MMD) operator [Gretton et al., 2012] which allows to compare distributions by embedding them in a feature space and comparing the mean of the respective embeddings, we propose to tackle the point classification task by training a network to distinguish between classes based on a feature embedding related to the first and second order statistics of the respective point's neighborhood. This embedding contains enough information for the classification network to learn the difference between non-edge, sharp-edge, and boundary points while at the same time being well structured and compact, making our solution very fast in terms of processing time. Various results of our *Boundary and Edge Detection* (BoundED) approach are depicted in Figure 5.1.

Our main contributions can be summarized as follows:

- We present a novel set of features for edge and boundary characterization and detection capturing local neighborhood information of point clouds better and being cheaper to compute than state-of-the-art approaches [Himeur et al., 2021].
- We demonstrate the benefits of this novel feature embedding at the example of a modified state-of-the-art neural edge detection network architecture giving better results with an even smaller network.

- Our evaluation demonstrates the ability of the proposed features to capture information regarding boundary classification of points in addition to edge classification.

5.3 Related Work

The detection of 3D edges in terms of sharp features, feature contours, or curves within unstructured point cloud data is a challenging task. Conventional methods include surface mesh reconstruction or graph-based approaches and analyzing local neighborhoods of each individual point based on principal component analysis (PCA). Thereby, the given connectivity information of a point with respect to its neighbors allows for a faster nearest neighbor search in comparison to unstructured point sets. However, preserving sharp edges and complex features in a reconstructed model is challenging due to smoothing effects induced by several reconstruction techniques. Directly extracting edges from unstructured point clouds has been addressed based on computing geometric descriptors per point based on the local covariance characteristics [Gumhold et al., 2001; Gelfand and Guibas, 2004]. Respective variants include taking the ratio between the Eigenvalues of the local covariance matrices on a single scale [Mérigot et al., 2011; Xia and Wang, 2017] or different scales [Pauly et al., 2003; Bazazian et al., 2015], local slippage analysis to define edges between segments of rotationally and translationally symmetrical shapes such as planes, spheres, and cylinders [Gelfand and Guibas, 2004], or directly estimating curvature [Lin et al., 2015; Nguyen et al., 2018]. Considering multiple scales reduces the susceptibility to noise, but such methods still rely on the suitable specification of a decision threshold. Non-parametric edge extraction has been achieved via kernel regression [Öztireli et al., 2009] or Eigenvalue analysis [Bazazian et al., 2015]. Others focused on detecting depth-discontinuities based on finding triangles with oblique orientations or finding triangles with long edges [Tang et al., 2007] or focusing on high-curvature points given as the extremum of curvatures [Fan et al., 1987] or curvature-guided region growing [Rusu et al., 2008]. In addition, edge detection has been approached based on normal variation analysis [Che and Olsen, 2018], 3D Canny edge detection [Monga et al., 1991], the combination of normal estimation and graph theory [Yagüe-Fabra et al., 2013], alpha-shapes [Edelsbrunner and Mücke, 1994], or boundary detection via DBSCAN-based detection and segmentation of 3D planes [Chen et al., 2022a].

Further approaches followed a moving least-squares (MLS) surface reconstruction with the subsequent detection of 3D edges based on a Gaussian map clustering computed within a local neighborhood [Demarsin et al., 2007; Weber et al., 2010; Weber et al., 2012; Ni et al., 2016]. The consideration of higher-order local approximations of non-oriented input gradients in MLS-based reconstruction has been used for the computation of continuous non-oriented gradient fields [Chen et al., 2013b], which allows a better preservation of surface or image structures. Another possibility to achieve continuously differentiable surfaces consists in exploring the scale-space for MLS [Mellado et al., 2012]. Furthermore, a scalar *sharpness* field defined on the underlying Moving Least-Squares surface of the point cloud has been proposed, where local maxima correspond to sharp edges [Raina et al., 2018; Raina et al., 2019]. Other approaches include the combination of adaptive reconstruction kernels [Fleishman et al., 2005] and spline fitting [Daniels Ii et al., 2008], the detection of boundary points and internal points as well as the subsequent application of a Fast-Fourier-Transform-based edge reconstruction to avoid the need to define a specific order for polynomial curve fitting [Mineo et al., 2019], the use of subspace detection and feature intersection [Fernandes and Oliveira, 2012], mean-shift-based selection of the most distant points with respect to the centroid of their neighborhood [Ahmed et al., 2018], the use of locally defined curve set features [Li and Hashimoto, 2017], the intersection of automatically detected planes [Mitropoulou and Georgopoulos, 2019], the filtering of potential feature points according to their local topology graph based on binary patterns [Guo et al., 2022], or RANSAC-based spatial regularization of sharp feature detector responses [Lin et al., 2015]. In addition, gradient-based edge detection with a subsequent non-maxima suppression and edge linking into linear and smooth structures [Xia and Wang, 2017] has been investigated.

Along the rapid progress in machine learning, learning-based approaches have been proposed for classifying individual points as *edge* or *non-edge*. Besides approaches based on least square regression or support vector machines [Wang et al., 2019b] that, however, had not been investigated in a general scenario, this can be achieved by the use of multi-scale features with a random forest based edge classification [Hackel et al., 2016a] or neural network based edge classifier [Himeur et al., 2021]. Other approaches include the neural network based pointwise distance estimation to the next sharp geometric feature [Matveev et al., 2022], or neural network based edge-aware point set consolidation [Yu et al., 2018] and 3D semantic edge detection based on a two-stream fully-convolutional network to jointly perform edge detection and semantic segmentation [Hu et al., 2020]. A further method [Wang et al., 2020] trains two neural networks to classify points into corners and edges based on a PointNet++ like architecture [Qi et al., 2017b]. After a subsequent non-maxima suppression of the classified points and their PointNet++ based clustering, a two-headed PointNet [Qi et al., 2017a] generates the final set of curves. This concatenation of deep networks induces a high computational burden and relies on high resource requirements. In addition, the learning of multi-scale local shape properties (e.g., normal and curvature) [Guerrero et al., 2018] and the use of CNNs for adaptive feature extraction from observations in a camera and laser-scanner setup [Xiao et al., 2019] have been investigated. Furthermore, the prediction of part boundaries within a 3D point cloud based on a graph convolutional network has been proposed [Loizou et al., 2020]. Further purely on boundary detection focused methods include the initial extraction of the exterior boundary based on neighborhood characteristics and the subsequent analysis regarding whether a point belongs to a hole boundary [Nguyen et al., 2015], and approaches based on a deep neural network [Tabib et al., 2020].

There are also a few image-based approaches that initially convert the 3D point cloud data into images [Lin et al., 2015]. Subsequently, a line segment detector [Von Gioi et al., 2008] is used to extract lines in 2D, which are backprojected to the point cloud. Another approach [Lu et al., 2019] relies on an initial segmentation of the point cloud into planar regions based on region growing and merging, which is followed by a plane-wise point projection into a 2D image and a final 2D contour extraction and backprojection to get the respective line segment in 3D space.

With our approach we follow the avenue of neural network based edge and boundary

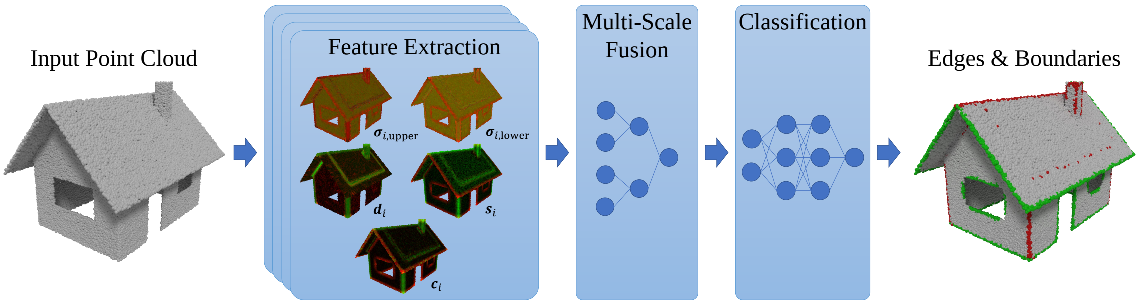

Figure 5.2: Overview of our BoundED approach: Based on an input point cloud, several features describing the local geometry are extracted on multiple scales. After pairwise fusion of the features for different scales, we classify the input points by an MLP leveraging the fused features as either non-edge, sharp-edge, or boundary points.

detection within 3D point clouds. We take inspiration from the *maximum mean discrepancy* (MMD) operator [Gretton et al., 2012] for the definition of a local feature embedding with respect to first- and second-order statistics of a local point's neighborhood, and we show that this embedding allows robust detection of non-edge, sharp-edge, and boundary points already with a compact network, thereby enabling fast inference times.

5.4 Methodology

With our approach, that we denote as BoundED, we aim at the robust and fast detection of non-edge, sharp-edge, or boundary points within given point clouds. For this purpose, we leverage the combination of a local encoding of feature characteristics based on the maximum mean discrepancy operator with respect to the local first- and second-order statistics and their efficient classification based on a compact multi-layer perceptron (MLP) (see Figure 5.2). In the following sections, we provide detailed descriptions regarding these aspects as well as respective implementation details.

5.4.1 Feature Computation

To compute meaningful features as input for the consecutive neural classification step, we generalize the idea of dividing a set of 3D points into two disjoint subsets and analyzing their respective covariances introduced by Bode et al. [2022] in the context of image denoising.

Let $\mathcal{P} = \{\boldsymbol{p}_i\}$ for $i = 1, \ldots, n$ be the given 3D point cloud consisting of n points. Using the k-nearest neighbors (k-NN) operator $\text{NN}_k(\boldsymbol{p}, \mathcal{P})$, we extract local neighborhoods

$$\mathcal{N}_{i,k} = \text{NN}_k(\boldsymbol{p}_i, \mathcal{P}) \tag{5.1}$$

with k points each. Throughout the remainder of this section, the neighborhood size k is omitted for notational simplicity.

For sufficiently dense point clouds in the absence of noise, this neighborhood represents a roughly disc-shaped set of points. In order to be invariant to the scale and sampling of the given point cloud, the point sets are normalized individually before features can be extracted. We propose to utilize the covariance matrix $\boldsymbol{K}_i = \operatorname{cov}(\mathcal{N}_i)$ for this purpose. By conducting an SVD of the covariance matrix

$$\boldsymbol{K}_i = \boldsymbol{U}_i \boldsymbol{\Sigma}_i \boldsymbol{V}_i^T \tag{5.2}$$

singular values $\sigma_{i,j} = \boldsymbol{\Sigma}_{i,jj}$ can be read from the diagonal entries of the matrix $\boldsymbol{\Sigma}_i$. Without loss of generality, these sigular values are assumed to be sorted in descending order, i.e. $\sigma_{i,1} \geq \sigma_{i,2} \geq \sigma_{i,3}$. Intuitively, these singular values are directional variances with directions being given by the corresponding Eigenvectors. Since $\mathcal{N}_i$ is roughly disk shaped, $\sigma_{i,1}$ and $\sigma_{i,2}$ can be seen as variance in direction of the disk's perpendicular semiaxes. Note, that in general $\sigma_{i,1}$ and $\sigma_{i,2}$ are similar but not equal as the points $\mathcal{N}_i$ will never represent a perfect uniformly sampled disk in practice. For the purpose of normalization, the neighborhood is centered around the origin according to the neighborhood's center of mass

$$\bar{\mathcal{N}}_i = \frac{1}{|\mathcal{N}_i|} \sum_{\boldsymbol{p} \in \mathcal{N}_i} \boldsymbol{p} \tag{5.3}$$

and scaled by the average standard deviation along the semiaxes:

$$\hat{\mathcal{N}}_i = \left\{ \frac{2}{\sqrt{\sigma_{i,1}} + \sqrt{\sigma_{i,2}}} (\boldsymbol{p} - \bar{\mathcal{N}}_i) \mid \boldsymbol{p} \in \mathcal{N}_i \right\}. \tag{5.4}$$

Besides normalization of the neighborhood, this SVD and in particular the Eigenvector $\boldsymbol{n}_i$ corresponding to $\sigma_{i,3}$ is utilized for further processing as this vector together with the neighborhood's center of mass $\bar{\mathcal{N}}_i$ defines a least-squares fitted plane to $\mathcal{N}_i$. Note that, in contrast to other approaches like e.g. PCEDNet [Himeur et al., 2021], by using this Eigenvector $\boldsymbol{n}_i$ as normal, our BoundED does not rely on any precomputed normals but only on the 3D positions of the points. We have observed, that the orientation of $\boldsymbol{n}_i$ can be unstable near outliers. Thus, we only consider the $\lfloor k/2 \rfloor$ points closest to $\bar{\mathcal{N}}_i$ for this step. According to this plane, the neighborhood is partitioned into two disjoint subsets

$$\mathcal{N}_{i,\text{upper}} = \left\{ \boldsymbol{p} \in \hat{\mathcal{N}}_i \mid \langle \boldsymbol{p}, \boldsymbol{n}_i \rangle \geq 0 \right\} \tag{5.5}$$

$$\mathcal{N}_{i,\text{lower}} = \left\{ \boldsymbol{p} \in \hat{\mathcal{N}}_i \mid \langle \boldsymbol{p}, \boldsymbol{n}_i \rangle < 0 \right\}. \tag{5.6}$$

As depicted in Figure 5.3, an analysis of these provides valuable information regarding local geometry. We propose to analyze the subset's statistics to capture this information. In particular, singular values $\sigma_{i,\text{upper},j}$ and $\sigma_{i,\text{lower},j}$ for $j \in \{1, 2, 3\}$ are computed by means of individual SVDs of the covariance matrices of $\mathcal{N}_{i,\text{upper}}$ and $\mathcal{N}_{i,\text{lower}}$ respectively. Additionally,

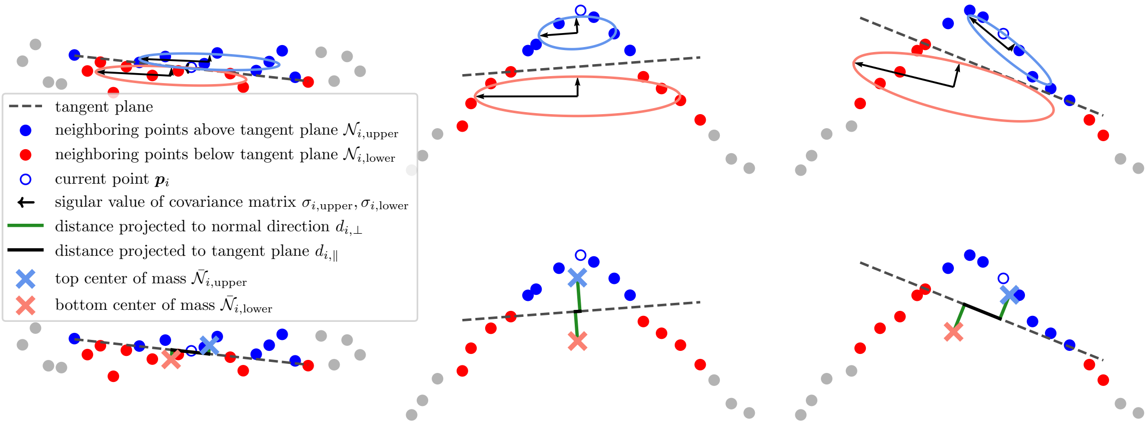
tangent plane
neighboring points above tangent plane $\mathcal{N}_{i,\text{upper}}$
neighboring points below tangent plane $\mathcal{N}_{i,\text{lower}}$
current point $\boldsymbol{p}_i$
sigular value of covariance matrix $\sigma_{i,\text{upper}}, \sigma_{i,\text{lower}}$
distance projected to normal direction $d_{i,\perp}$
distance projected to tangent plane $d_{i,\|}$
top center of mass $\bar{\mathcal{N}}_{i,\text{upper}}$
bottom center of mass $\bar{\mathcal{N}}_{i,\text{lower}}$

Figure 5.3: Features extracted from the local neighborhood of a point. Points can be classified as non-edge or sharp-edge by analyzing their neighborhood with respect to singular values and means of points above and below a least-squares fitted tangent plane. Planar neighborhoods (left) tend to have similar values for $\sigma_{i,\text{upper}}$ and $\sigma_{i,\text{lower}}$ while having low values for $d_{i,\perp}$. Sharp-edge neighborhoods (middle) exhibit a larger difference in $\sigma_{i,\text{upper}}$ and $\sigma_{i,\text{lower}}$ as well as large $d_{i,\perp}$. In contrast, neighborhoods of points close to sharp-edges (right) have higher $d_{i,\|}$ than neighborhoods of points directly on the edge.

the distance between the centers of mass of both subsets

$$\bar{\mathcal{N}}_{i,\text{upper}} = \frac{1}{|\mathcal{N}_{i,\text{upper}}|} \sum_{\boldsymbol{p} \in \mathcal{N}_{i,\text{upper}}} \boldsymbol{p} \tag{5.7}$$

$$\bar{\mathcal{N}}_{i,\text{lower}} = \frac{1}{|\mathcal{N}_{i,\text{lower}}|} \sum_{\boldsymbol{p} \in \mathcal{N}_{i,\text{lower}}} \boldsymbol{p} \tag{5.8}$$

decomposed into perpendicular and tangential components is calculated as

$$d_{i,\perp} = \langle \bar{\mathcal{N}}_{i,\text{upper}} - \bar{\mathcal{N}}_{i,\text{lower}}, \boldsymbol{n}_i \rangle \tag{5.9}$$

$$d_{i,\|} = \|(\bar{\mathcal{N}}_{i,\text{upper}} - \bar{\mathcal{N}}_{i,\text{lower}}) - d_{\perp} \boldsymbol{n}_i\|_2. \tag{5.10}$$

Intuitively, low values for $d_{i,\perp}$ indicate that the local neighborhood $\mathcal{N}_i$ is near planar and thus the probability for $\boldsymbol{p}_i$ being part of a sharp edge is small. In contrast, high values are found in areas with a high amount of geometric detail or noise. A large tangential distance $d_{i,\|}$ can indicate, that an edge is close-by, but $\boldsymbol{p}_i$ may not necessarily be coincident (see Figure 5.3).

Furthermore, inspired by Bendels et al. [2006], to improve detection of outliers and boundaries, the perpendicular and tangential components of the distance between $\boldsymbol{p}_i$ and the center of mass of its k nearest neighbors are computed as:

$$s_{i,\perp} = \langle \boldsymbol{p}_i - \bar{\mathcal{N}}_i, \boldsymbol{n}_i \rangle \tag{5.11}$$

$$s_{i,\|} = \|(\boldsymbol{p}_i - \bar{\mathcal{N}}_i) - s_{\perp} \boldsymbol{n}_i\|_2. \tag{5.12}$$

While not necessarily always following this observation, points at boundaries tend to have large $s_{i,\|}$ and at the same time small $s_{i,\perp}$. Intuitively, the neighbors of points at boundaries are all on one side which indicates that they are far away from the center of mass of their neighborhood. If $\boldsymbol{p}_i$ is an outlier near a well-defined surface, the corresponding $s_{i,\perp}$ tends to be large.

In summary, the analysis yields the following features: the singular values

$$\boldsymbol{\sigma}_{i,\cdot} = (\sigma_{i,\cdot,1}, \sigma_{i,\cdot,2}, \sigma_{i,\cdot,3})^T \tag{5.13}$$

of the upper and lower subsets respectively, the perpendicular and tangential distances between the centers of mass of both subsets

$$\boldsymbol{d}_i = (d_{i,\perp}, d_{i,\|})^T, \tag{5.14}$$

and the perpendicular and tangential distances between the point $\boldsymbol{p}_i$ and the center of mass of its neighborhood

$$\boldsymbol{s}_i = (s_{i,\perp}, s_{i,\|})^T. \tag{5.15}$$

Thus, we assemble a per-point 10D feature vector according to

$$\hat{\boldsymbol{x}}_i = (\boldsymbol{\sigma}_{i,\text{upper}}, \boldsymbol{\sigma}_{i,\text{lower}}, \boldsymbol{d}_i, \boldsymbol{s}_i)^T. \tag{5.16}$$

5.4.2 Multi-Scale Feature Embedding

In order to classify points $\boldsymbol{p}_i$ of a point cloud $\mathcal{P}$ as non-edge, sharp-edge, or boundary, the per-point data $\boldsymbol{X}_i$ is individually processed by a small MLP. $\boldsymbol{X}_i$ relies on computing $\hat{\boldsymbol{x}}_{i,k}$ on m different scales $k_0, \ldots, k_{(m-1)}$, i.e. choosing neighborhoods containing varying numbers of points k, for each point $\boldsymbol{p}_i$. Inspired by the GLS [Mellado et al., 2012] features utilized by PCEDNet [Himeur et al., 2021], we add the tangential and perpendicular distances

$$c_{i,k,\perp} = \langle \bar{\mathcal{N}}_{i,k} - \overline{(\mathcal{N}_{i,k_0} - \mathcal{N}_{i,k})}, \boldsymbol{n}_{i,k_0} \rangle \tag{5.17}$$

$$c_{i,k,\|} = \|(\bar{\mathcal{N}}_{i,k} - \overline{(\mathcal{N}_{i,k_0} - \mathcal{N}_{i,k})}) - c_{i,k,\perp}\boldsymbol{n}_{i,k_0}\|_2. \tag{5.18}$$

between the center of mass $\bar{\mathcal{N}}_{i,k}$ of each scale k and the center of mass of points of the largest scale's neighborhood $\mathcal{N}_{i,k_0}$ which are not part of $\mathcal{N}_{i,k}$ as well to each $\hat{\boldsymbol{x}}_{i,k}$:

$$\boldsymbol{x}_{i,k} = (\hat{\boldsymbol{x}}_{i,k}, \boldsymbol{c}_{i,k})^T, \tag{5.19}$$

where

$$\boldsymbol{c}_{i,k} = (c_{i,k,\perp}, c_{i,k,\|})^T. \tag{5.20}$$

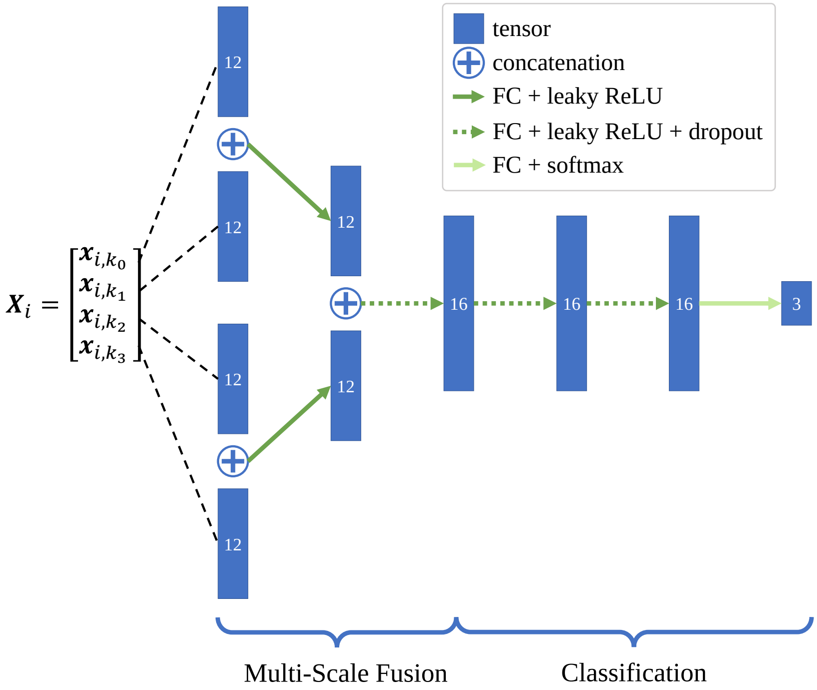
tensor
concatenation
FC + leaky ReLU
FC + leaky ReLU + dropout
FC + softmax
$X_i = \begin{bmatrix} x_{i,k_0} \\ x_{i,k_1} \\ x_{i,k_2} \\ x_{i,k_3} \end{bmatrix}$
12
12
12
12
12
12
16
16
16
3
Multi-Scale Fusion
Classification

Figure 5.4: Architecture of the multi-scale fusion and classification network consisting of fully connected (FC) layers, leaky rectified linear unit (leaky ReLU) activations, dropout, and softmax function. Features computed on multiple scales are combined in a pairwise manner and afterwards processed by an MLP to classify a point as non-edge, sharp-edge, or boundary.

The complete multi-scale per-point features can be written in matrix form as

$$\boldsymbol{X}_i = \begin{bmatrix} x_{i,k_0,1} & \cdots & x_{i,k_0,12} \\ \vdots & \ddots & \vdots \\ x_{i,k_{(m-1)},1} & \cdots & x_{i,k_{(m-1)},12} \end{bmatrix}. \tag{5.21}$$

These multi-scale features are fused in a pair-wise manner similarly to PCEDNet [Himeur et al., 2021] as depicted in Figure 5.4, before being processed by the classification MLP itself.

5.4.3 Network Architecture

For our experiments, we use features computed on four different scales using 128, 64, 32, and 16 neighboring points respectively. In contrast to PCEDNet, BoundED uses less scales, i.e. 4 instead of 16. However, to accomodate for the lost network depth due to using less scales, an additional hidden layer is added to the classification MLP, giving the network a total of 1.6k learnable parameters. For training the network, a focal loss [Lin et al., 2017] with $\gamma = 2$ is used as training batches are usually very unbalanced due to the small number of edge points compared to non-edge points in most point clouds. Furthermore, we propose to add dropout [Srivastava et al., 2014] with $p = 0.5$ to the classification layers to prevent overfitting and facilitate a more stable training process.

5.4.4 Implementation Details

Our algorithm is implemented using PyTorch [Paszke et al., 2019] for feature extraction as well as the neural network and its training. For finding the local neighborhood of points, the k-NN implementation of PyTorch3D [Ravi et al., 2020] is used. Due to the point sets $\mathcal{N}_{i,k,\text{upper}}$, $\mathcal{N}_{i,k,\text{lower}}$ containing different numbers of points for different $\boldsymbol{p}_i$, we employ masking to efficiently vectorize the task and fully utilize the tremendous computation capabilities of modern GPUs during the feature extraction phase. The network is trained using the Adam optimizer [Kingma and Ba, 2014] with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and learning rate 0.001. Batch size is set to 16384. The number of training iterations varies between used datasets and is described in detail in Section 5.5.1.

5.5 Results and Discussion

In the following, the effectiveness of the proposed combination of our novel multi-scale features and our compact classification network is evaluated quantitatively as well as qualitatively on several different datasets. We focus mostly on the comparison with the state-of-the-art point cloud edge detection network PCEDNet by Himeur et al. [2021] as it is the most relevant previous work due to also being designed to be fast and compact. Similar to our BoundED approach, they rely on feeding their classification network with multi-scale per-point features allowing for a direct comparison of the used embeddings. Furthermore, boundary detection capabilities of our network are assessed. Finally, an experiment on noisy data as well as an ablation study regarding the chosen features and the employed number of scales further validate our results.

5.5.1 Datasets

We train and evaluate our approach on several different datasets and provide comparisons to other point cloud edge detection algorithms. To allow for a direct comparison with PCEDNet [Himeur et al., 2021], their *Default* dataset as well as the publicly available *ABC* [Koch et al., 2019] dataset are used.

Default Introduced by Himeur et al. [2021], this dataset is designed to be as small as possible in order to facilitate very short training times with only a few simple hand-labeled point clouds to train on but still generalize well to arbitrary other point clouds. It contains 9 point clouds for training as well as 7 different point clouds for evaluation. To form the validation set, 1000 points are randomly sampled from each class. Despite containing three different classes of points, i.e. non-edge, sharp-edge, and smooth-edge, originally, this work

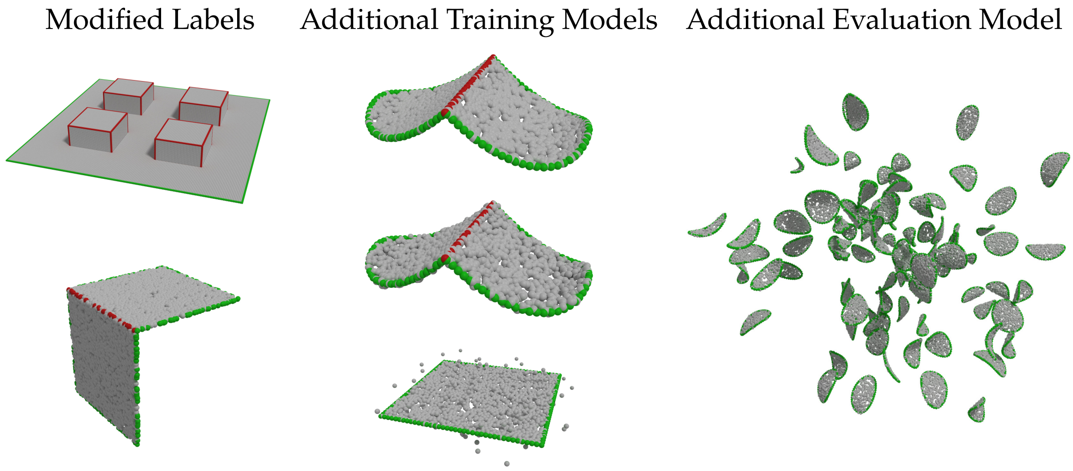

Figure 5.5: Adjustments made to the *Default* dataset to facilitate boundary detection. The labels of two point clouds (left) are modified to identify already included boundary points correctly. Three simple point clouds (middle) are added to the training and validation set to improve the coverage of potential boundary point cases. For an additional evaluation we add a further point cloud (right) to ensure a sufficient representation of boundary points in the evaluation data.

focuses on non-edge and sharp-edge classification only and therefore treats smooth-edge points as non-edge points in all results. We train BoundED for 3000 iterations on this dataset.

ABC The *ABC* dataset published by Koch et al. [2019] is a very large collection of CAD models accompanied with triangle meshes and feature annotations among other data. Point clouds are generated from triangle meshes by simply removing all edges and faces. A ground truth classification label for each point is extracted by checking whether it is part of any CAD curve flagged as *sharp*. To ensure a meaningful comparison with the work by Himeur et al. [2021], we also only use chunk 0000 and exactly the same 200 models for training and 50 models for validation while also using all 7168 point clouds for evaluation. As *ABC* contains many more points than *Default*, we train our network for 8000 iterations on its training data.

Default++ As the original *Default* dataset published by Himeur et al. [2021] does not include annotated boundary vertices, which prevents its use for training models for boundary detection tasks, we propose to extend it as shown in Figure 5.5 to create the *Default*++ dataset. The original *Default* dataset contains two models containing boundary points not annotated as such. Thus, the first modification is to add these boundary annotations accordingly. Furthermore, it is extended by two additional point clouds for training, which were specifically designed to contain clean and noisy curved boundaries of varying radii as these cases are not included in the original *Default* dataset. Finally, to prevent boundary points from being heavily underrepresented in the evaluation set, we additionally add an evaluation model containing a multitude of boundary situations with varying levels of noise.

Since the class of boundary points in the training set is still much smaller than the classes of non-edge or sharp-edge points, we only add 100 randomly sampled boundary points to the validation set. The resulting training set contains 279.5k non-edge, 15.7k sharp-edge, and only 0.9k boundary points. As it is similar in size to the *Default* dataset, we use the same 3000 iterations to train on *Default++*.

Additional Evaluation Data To assess the capabilities of the proposed algorithm more thoroughly, we also use publicly available point clouds of 3D scanned buildings and plants. The *christ_church*[1] point cloud contains 1.9 million points of the Christ Church Cathedral and its surrounding in Dublin. Furthermore, the *pisa_cathedral*[2] point cloud with 2.5 million points scanned by Mellado et al. [2015] is used as well. The *station*[3] point cloud is an even larger point cloud representing a train station as 12.5 million points which we also use for evaluation. Finally, we are using point clouds of three different plants scanned by Conn et al. [2017]: An Arabidopsis[4], a Tobacco[5], and a Tomato[6] plant with 172k, 1474k, and 226k points respectively.

5.5.2 Metrics

Similarly to the work by Himeur et al. [2021], we use several metrics for comparison: Precision, Recall, Matthews Correlation Coefficient (MCC), F1 score, Accuracy, and Intersection over Union (IoU, also known as Jaccard index). Precision evaluates the ratio of true classifications as sharp-edge or boundary to the total number of classified points. In contrast, Recall measures the ratio of correctly classified sharp-edge or boundary points compared to the true number of such points existing in the processed model. Precision and Recall are coupled, i.e. Precision increases and Recall decreases if only points exhiting very high confidence are classified and vice versa. Thus, mainly the other mentioned metrics, which combine Precision and Recall scores in different ways, are used for directly comparing our BoundED technique to related works.

[1] Available at: `https://sketchfab.com/3d-models/christ-church-and-dublin-city-councilb5f6bcce8ebc44a3b4bbb6b0fef067b3`, accessed on 10/14/2022.

[2] Available at: `https://www.irit.fr/recherches/STORM/MelladoNicolas/category/datasets/`, accessed on 10/22/2022.

[3] Available at: `https://sketchfab.com/3d-models/station-rer-6c636ca4793345e8ae12beb97b7d6359`, accessed on 10/14/2022.

[4] Available at: `http://plant3d.navlakhalab.net/shoots/public/view/plant/40`, time point 33, accessed on 10/14/2022.

[5] Available at: `http://plant3d.navlakhalab.net/shoots/public/view/plant/20`, time point 30, accessed on 10/14/2022.

[6] Available at: `http://plant3d.navlakhalab.net/shoots/public/view/plant/15`, time point 30, accessed on 10/14/2022.

5.5.3 Comparison to Related Work

Throughout this section, we compare the performance of our work with the performance of several other recent related works for point cloud edge detection: Covariance Analysis (CA) [Bazazian et al., 2015], Feature Edges Estimation (FEE) [Mérigot et al., 2011; Alliez et al., 2022], ECNet [Yu et al., 2018], PIE-NET [Wang et al., 2020], PCPNet [Guerrero et al., 2018], and PCEDNet [Himeur et al., 2021]. The postfix *-2c* denotes that the respective algorithm has been trained for classification of two classes only, i.e. non-edge and sharp-edge, despite being originally designed to potentially handle more than two classes. For the quantitative evaluation (see Section 5.5.4), data reported by Himeur et al. [2021] is used for PCEDNet and PCPNet, while we use the numbers published by Wang et al. [2020] for ECNet and PIE-Net. For the two non-learning methods CA and FEE, we use one set of parameters each per dataset finetuned on the dataset's characteristics, i.e. more aggressive thresholding on clean data compared to noisy data: CA finetuned on *Default* uses 0.025 as threshold, while using 0.08 on *ABC*. The parameters for FEE are set to $R = 0.1$, $r = 0.03$ to work well with the *Default* dataset and to $R = 0.02$ and $r = 0.002$ to yield good results on the *ABC* dataset. In both cases, we use 0.16 as threshold. For FEE, we additionally normalize all point clouds to fit inside an axis-aligned unit box as R and r are related to the expected feature size, which varies heavily for the models in the *ABC* dataset. The PCEDNet results shown for the purpose of qualitative evaluation in Section 5.5.5 are generated using the publicly available precompiled demo application[7]. We assume only the point positions to be given as input for the algorithm. Since PCEDNet relies on point normals, these are generated according to the authors' specification using Meshlab [Cignoni et al., 2008]. To be able to report meaningful numbers for the quantitative evaluation in Section 5.5.4, we have done every experiment five times, evaluated the loss function over the validation set, and chose the best result according to this metric.

To ensure practicality of our algorithm, timings are reported for two different hardware configurations: On the one hand, we use an old consumer-grade Nvidia RTX 2080 Ti GPU with 11GB memory and an AMD Ryzen 3600X CPU with 32GB memory. On the other hand, we also used the recent enterprise Nvidia A40 GPU with 48GB memory and two AMD EPYC 7313 CPUs with 32 threads each and 512GB memory. Note, however, that we only used 12 worker threads in the data loader during training for both hardware configurations. We exclude the IO and network initialization time from the timings listed in this section and focus on reporting the time required by the actual feature extraction as well as network inference instead.

5.5.4 Quantitative Comparison

Tables 5.1 and 5.2 show median scores of various commonly used metrics to allow a quantitative comparison of our approach with others. For all experiments in this section, we are working on the *Default* and *ABC* datasets and aim at distinguishing sharp-edge points from non-edge points.

[7] Available at: `https://storm-irit.github.io/pcednet-supp/software.html`, accessed on 10/14/2022.

	Precision(↑)	Recall(↑)	MCC(↑)	F1(↑)	Accuracy(↑)	IoU(↑)
CA (Default) [Bazazian et al., 2015]	0.184	**0.891**	0.332	0.305	0.753	0.178
CA (ABC) [Bazazian et al., 2015]	0.183	0.357	0.188	0.242	0.863	0.138
FEE (Default) [Mérigot et al., 2011; Alliez et al., 2022]	0.241	0.866	0.400	0.376	0.828	0.232
FEE (ABC) [Mérigot et al., 2011; Alliez et al., 2022]	0.060	0.961	-0.021	0.113	0.082	0.060
PCEDNet-2c (*Default*) [Himeur et al., 2021]	0.364	0.611	0.402	0.430	0.908	0.274
BoundED (Ours) (*Default*)	**0.365**	0.595	**0.423**	**0.453**	**0.912**	**0.293**
BoundED (Ours) (*ABC*)	0.248	0.589	0.328	0.348	0.869	0.210

Table 5.1: Median scores of edge detection approaches evaluated on the *Default* dataset. The dataset used for parameter tuning or training is mentioned in parentheses. Data regarding PCEDNet-2c is taken from Himeur et al. [2021].

	Precision(↑)	Recall(↑)	MCC(↑)	F1(↑)	Accuracy(↑)	IoU(↑)
CA (Default) [Bazazian et al., 2015]	0.312	**0.991**	0.482	0.471	0.845	0.308
CA (ABC) [Bazazian et al., 2015]	0.498	0.820	0.541	0.574	0.929	0.403
FEE (Default) [Mérigot et al., 2011; Alliez et al., 2022]	0.178	0.621	0.213	0.270	0.775	0.156
FEE (ABC) [Mérigot et al., 2011; Alliez et al., 2022]	0.857	0.898	0.821	0.832	0.980	0.712
PCEDNet-2c (*Default*) [Himeur et al., 2021]	0.662	0.936	0.708	0.730	0.958	0.574
PCEDNet-2c (*ABC*) [Himeur et al., 2021]	0.735	0.984	0.808	0.822	0.970	0.597
ECNet (*ABC*) [Yu et al., 2018]	0.487	0.573	-	0.526	-	0.356
PIE-NET (*ABC*) [Wang et al., 2020]	0.692	0.858	-	0.766	-	0.622
PCPNet-2c (*ABC*) [Guerrero et al., 2018]	**0.954**	0.756	0.797	0.807	0.979	0.668
BoundED-2c (Ours) (*Default*)	0.420	0.594	0.381	0.420	0.909	0.266
BoundED-2c (Ours) (*ABC*)	0.932	0.833	**0.842**	**0.850**	**0.983**	**0.739**

Table 5.2: Median scores of edge detection approaches evaluated on the *ABC* dataset. The dataset used for training is mentioned in parentheses. Data regarding PCEDNet-2c, PCPNET-2c is taken from Himeur et al. [2021]. Data regarding ECNet, and PIE-NET is taken from Wang et al. [2020].

When training and evaluation are done on the *Default* dataset, our algorithm performs better than all related works in all metrics except for Recall, i.e. BoundED is not able to identify quite as many sharp-edge points as others, but more of those points classified as being a sharp-edge point are actually correctly identified as such. As we are also using a smaller network in comparison to PCEDNet, this suggests, that our multi-scale features are better at describing the geometry of the local neighborhood in terms of sharp edges than their GLS based features.

Also observe, that BoundED trained on *ABC* performs better than CA and FEE finetuned on *ABC* when evaluating on the *Default* dataset. Both non-learning approaches, i.e. CA and FFE, rely on setting a threshold to distinguish between sharp-edge and non-edge points. On clean data like the models from the *ABC* dataset, this threshold can be set much more aggressively. In the presence of noise, this, however, leads to the algorithms not detecting all edges in the case of CA and tremendous overclassification of points as sharp-edge points in the case of FEE.

When being evaluated on *ABC*, BoundED trained on *ABC* once again outperforms all other approaches in terms of MCC, F1, Accuracy, and IoU scores, but PCEDNet loses less effectiveness if being trained on *Default* in comparison to BoundED. While PCPNet has the highest Precision and CA trained on *Default* exhibits the highest Recall, they are worse in terms of overall classification performance due to having much worse scores in Recall and Precision respectively.

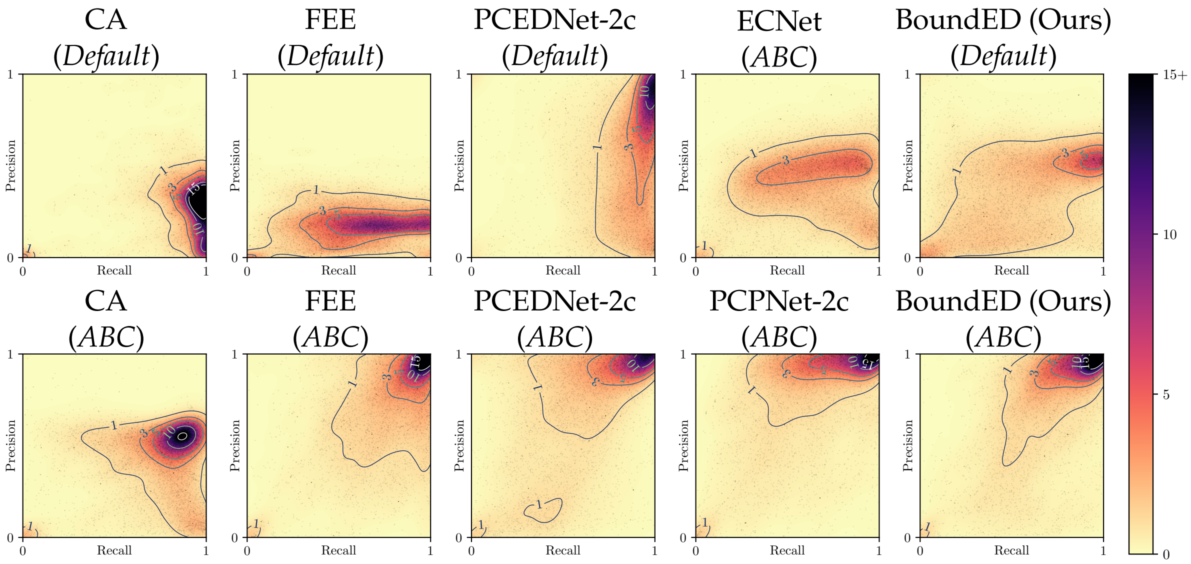

Figure 5.6: Precision-Recall-plots of most approaches listed in Table 5.2. Every small semi-transparent black dot corresponds to a single point cloud from the *ABC* dataset and its Precision and Recall scores when being processed by the respective approach. The background depicts the color-coded local density of points.

	Training		Evaluation	
	Preprocessing	Training	Preprocessing	Classification
PCEDNet (*Default*) [Himeur et al., 2021]	0:19 m	2:52 m	-	-
BoundED-2c (Ours) (*Default*, RTX 2080 Ti)	0:04 m	1:24 m	1.0 s	0.002 s
BoundED-2c (Ours) (*Default*, A40)	0:04 m	1:13 m	1.0 s	0.004 s
PCEDNet-2c (*ABC*) [Himeur et al., 2021]	2:11 m	20:00 m	2:35:00 h	0:25:30 h
BoundED-2c (Ours) (*ABC*, RTX 2080 Ti)	1:07 m	2:24 m	1:39:06 h	0:00:03 h
BoundED-2c (Ours) (*ABC*, A40)	0:59 m	3:00 m	1:20:17 h	0:00:04 h
BoundED (Ours) (*Default++*, RTX 2080 Ti)	0:05 m	1:22 m	1.4 s	0.002 s
BoundED (Ours) (*Default++*, A40)	0:04 m	1:13 m	1.4 s	0.008 s

Table 5.3: Comparison of time required to calculate the multi-scale features used as network input and training or evaluation time on the training or evaluation data respectively of the dataset in parentheses. Timings of our approach are determined on two different hardware configurations: An older consumer grade Nvidia RTX 2080 Ti GPU with 11GB memory and a recent enterprise grade Nvidia A40 GPU with 48GB memory. Data regarding PCEDNet and PCEDNet-2c is taken from Himeur et al. [2021].

The Precision-Recall-plots shown in Figure 5.6 confirm these observations. In these diagrams, every point cloud of the *ABC* dataset is depicted as one small semi-transparent black point according to it Precision and Recall scores. The background color depicts the color-coded local density of points. The plot for BoundED trained on the *ABC* dataset exhibits the highest density in the top right corner suggesting that the classification results on most models are of high quality, while the peak density for approaches trained or finetuned on *Default* is much lower and the individual points are more evenly distributed over a larger area.

Besides yielding better classification scores across the board, the computation of our features is also cheaper compared to PCEDNet and our multi-scale fusion and classification network has roughly 25% less parameters. Table 5.3 lists training and evaluation timings for PCEDNet and our approach. Training in this context consists of the multi-scale feature extraction for the training and validation data of the dataset given in parentheses as well as using this data to train the network. Similarly, evaluation consists of extracting the features on the evaluation set given in parentheses and classifying all points using the trained network.

While using a powerful GPU accelerates the feature extraction step, the difference for the network training and inference is negligible due to the networks compactness and simplicity.

5.5.5 Qualitative Comparison

If trained on *Default*++, our algorithm learns to identify the sharp edges in the evaluation models well as can be seen in Figure 5.7. The edge detection results seem to be even a bit more consistent than the ones of PCEDNet trained on *Default*. However, PCEDNet does a better job at classifying outliers as non-edge. We suspect, that the GLS features are more robust regarding outliers and our network was not able to compensate for this as outliers are strongly underrepresented in the training data.

Results on some evaluation models of the *ABC* dataset are depicted in Figure 5.8. PCEDNet exhibits mixed performance on the models 0027 and 0059. Depending on the dataset used for training, the algorithm either tends to have problems with the thin wall of model 0059 or produces less consistent results on some parts of model 0027. The most consistent results, however, are produced by our approach trained on the *Default*++ dataset. It is the only configuration that produces an inner circular edge on model 0027 without holes while not massively overclassifying the cylindrical wall at the top of the model as sharp edge. The classification results of points which are part of the screw thread in model 0117 are not as consistent as the detected sharp edges contain many holes. The screw is classified best by our algorithm trained on *ABC*, but this combination erroneously classifies the boundary of model 1222 as sharp edges.

BoundED also works well on actual 3D scanned real-world data as shown in Figures 5.9 and 5.10. On the *christ_church* point cloud, it outperforms PCEDNet in classifying the sharp-edges of roofs (see green zoom-in) and also gives good results for the fine stone structures of the church (see blue zoom-in). The results on the *station* point cloud are similar. Especially for the third row, our algorithm gives much more consistent results in the area of the escalator.

5.5.6 Boundary Detection

As already mentioned in Section 5.2, the processing of point clouds often requires the detection of boundaries in addition to sharp edges due to potentially very fine structures as well as finite resolution. This is especially important if the scanned object has many fine structures like leafs on plants or fine fins on buildings. Due to the GLS [Mellado et al., 2012]

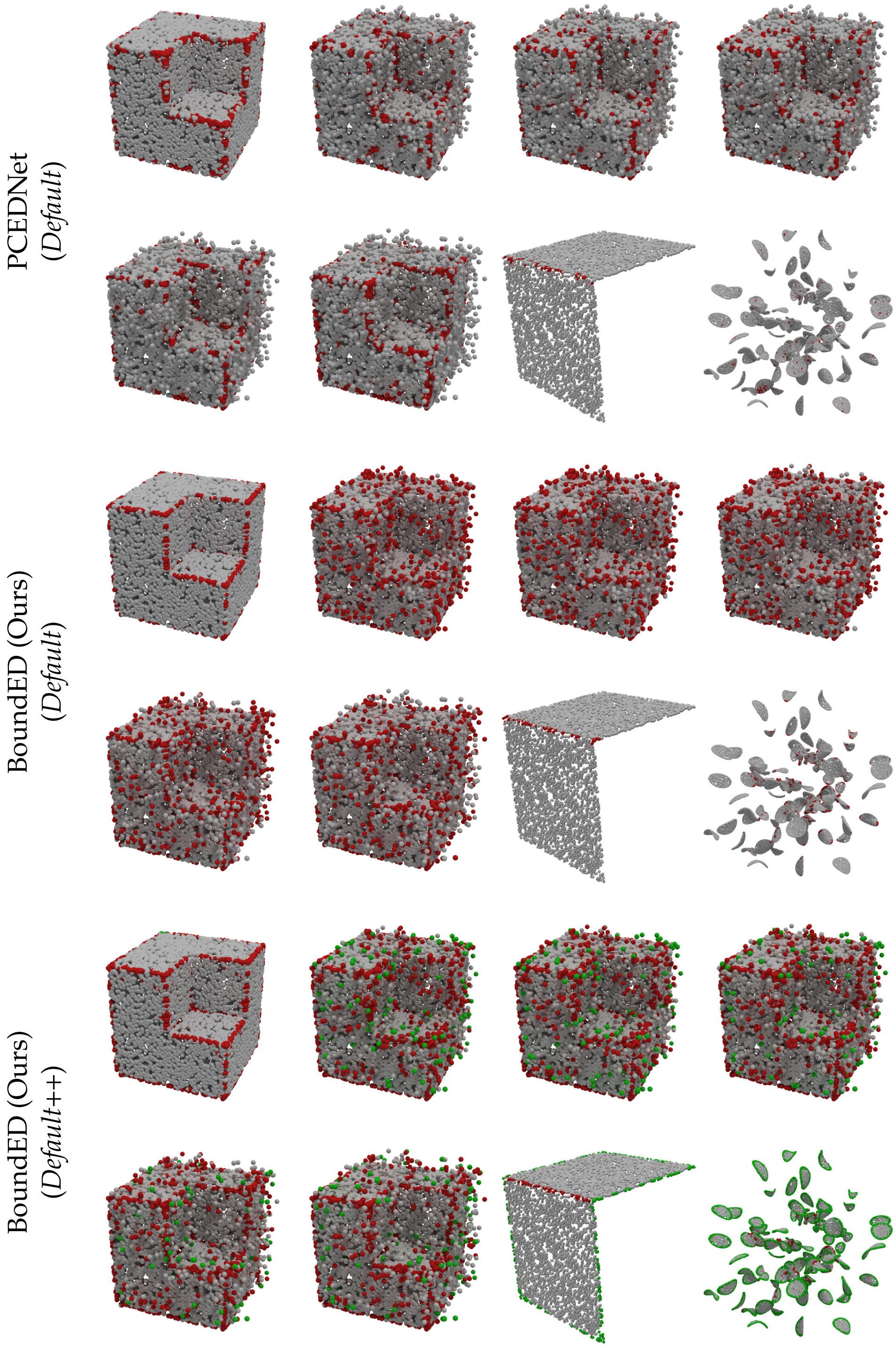

Figure 5.7: Comparison of the results on the *Default*++ evaluation set. The dataset used for training is reported in parentheses. As first and second row were trained on the *Default* dataset, the respective approches are by design not able to detect boundaries.

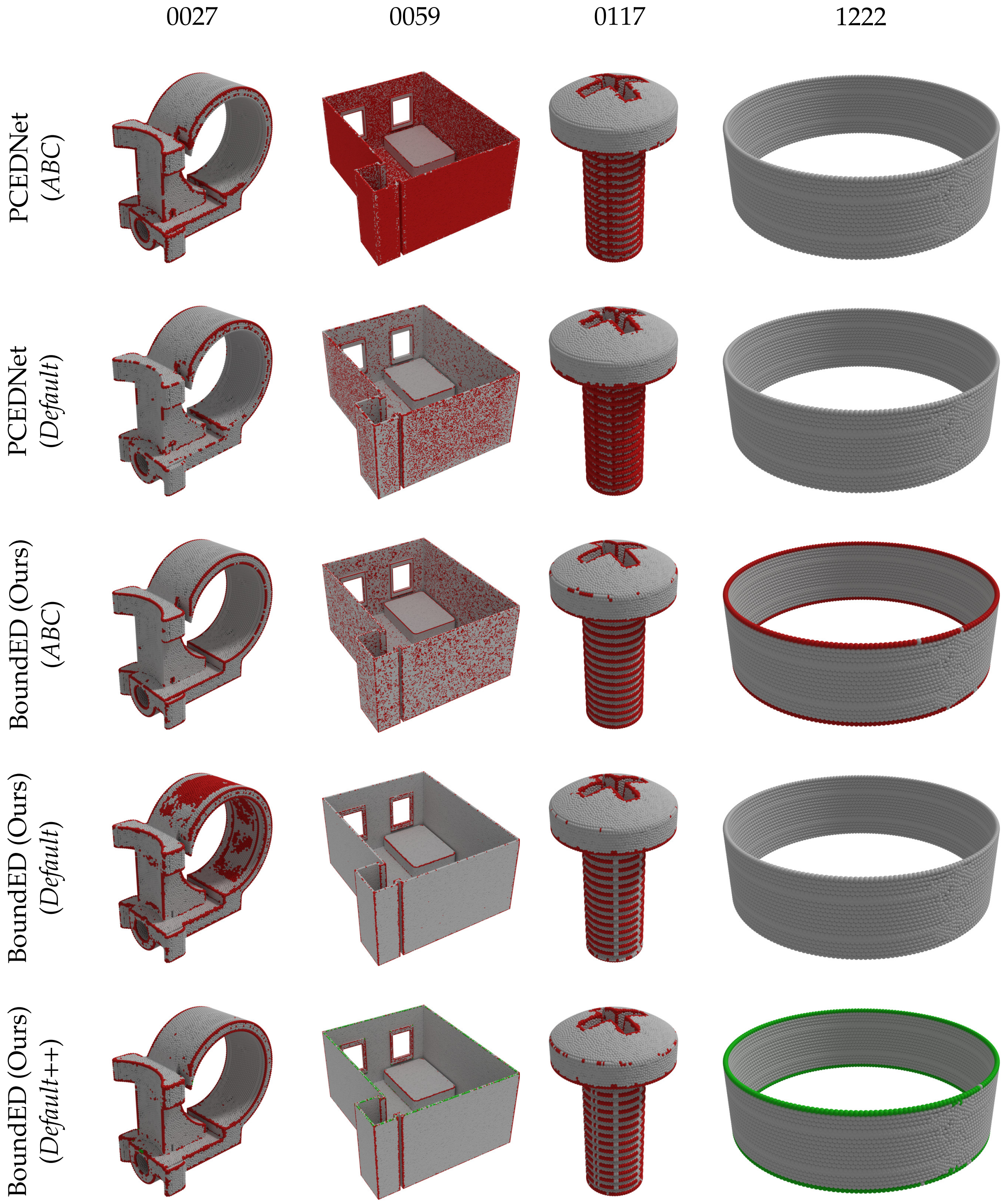

Figure 5.8: Comparison of PCEDNet and BoundED trained on three different datasets and evaluated on four different models from the *ABC* evaluation dataset. The dataset used for training the respective approach is given in parentheses. Algorithms trained on *Default* or *ABC* are not able to detect boundaries by design.

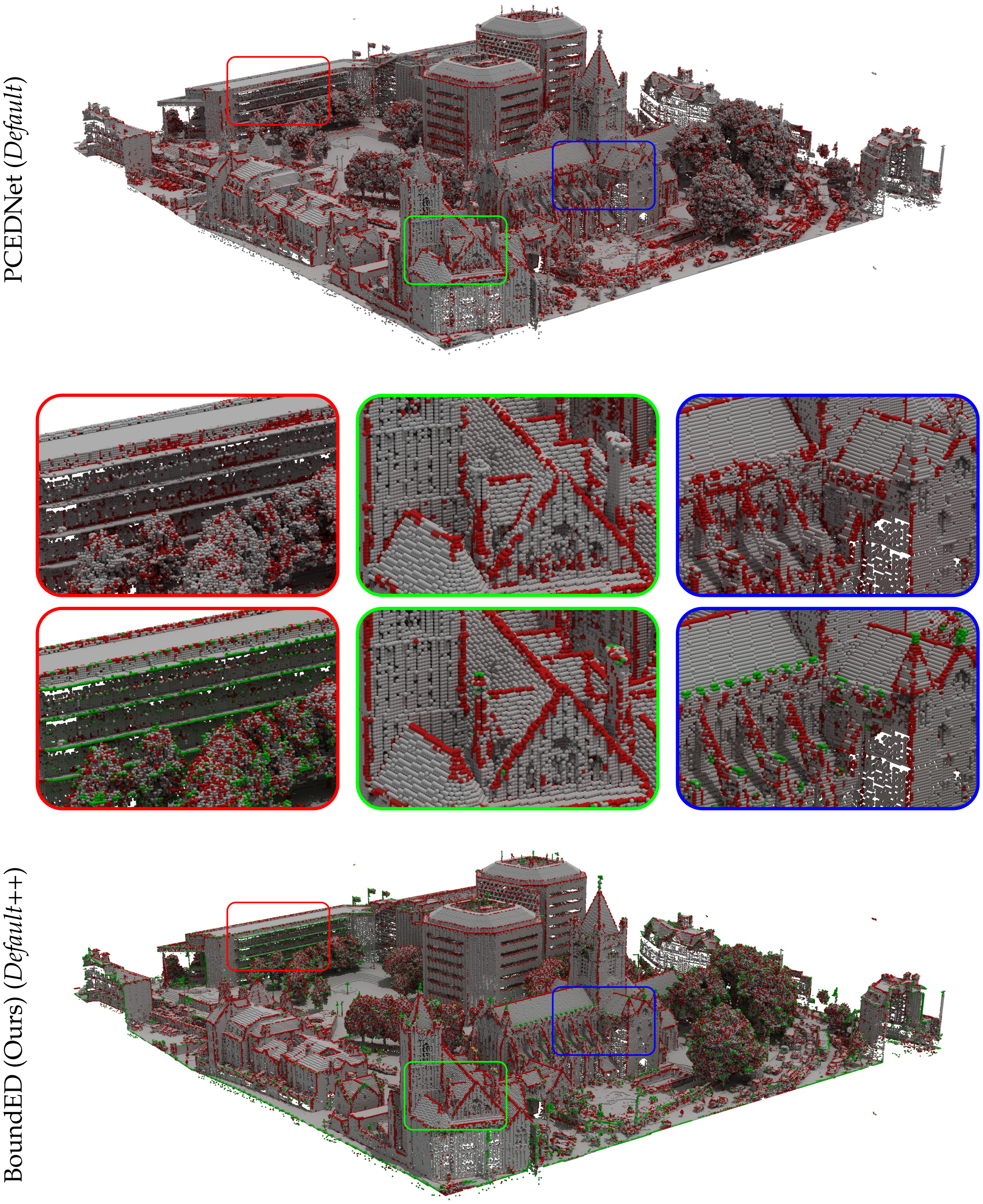

Figure 5.9: Classification result of PCEDNet trained on *Default* (top) and BoundED trained on *Default++* (bottom) for the mid-sized (1.9 million points) scanned *christ_church* point cloud. Three different zoomed parts are depicted for direct comparison (middle).

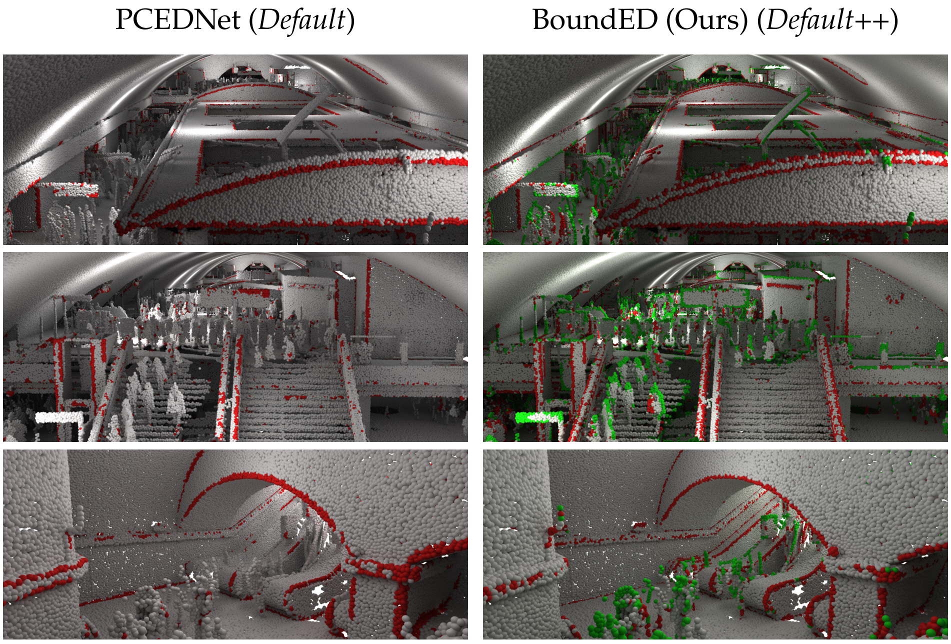

Figure 5.10: Classification result of PCEDNet trained on *Default* (left) and BoundED trained on *Default*++ (right) for the large (12.5 million points) scanned *train_station* point cloud.

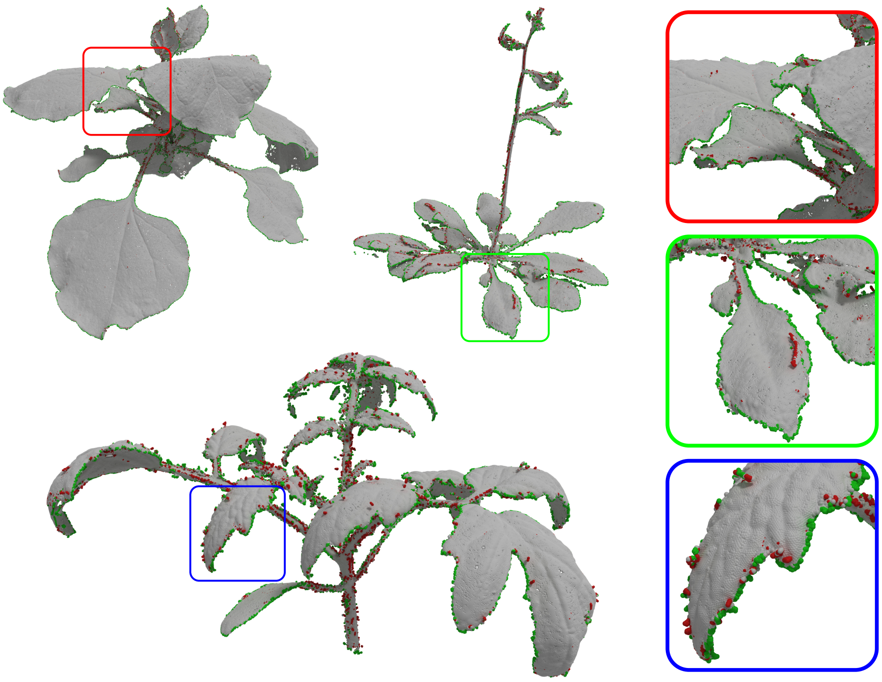

Figure 5.11: 3D scanned plant point clouds classified using our approach BoundED trained on *Default*++.

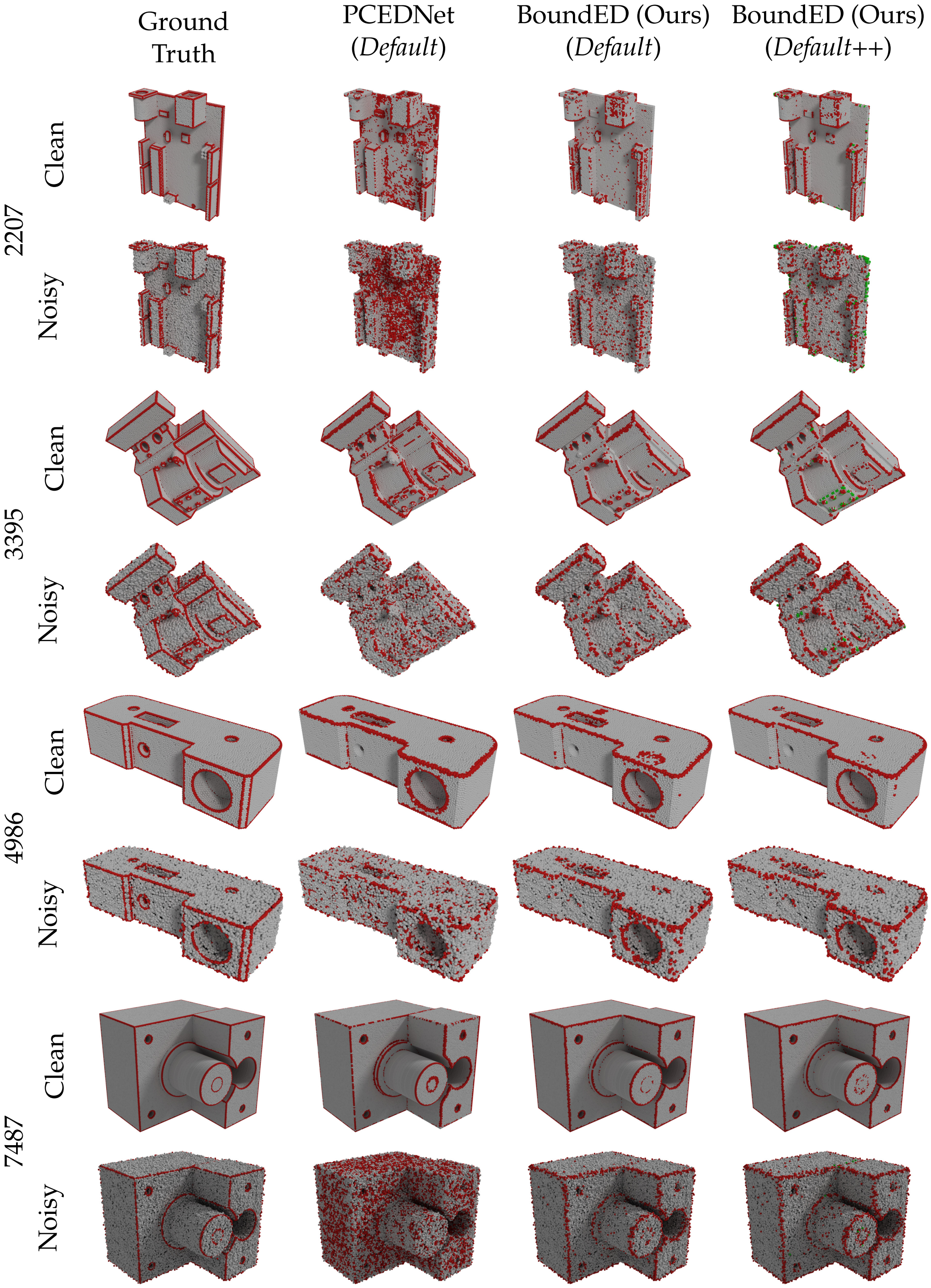

Figure 5.12: Comparison of PCEDNet and BoundED regarding bahavior on noisy data. The dataset used for training the respective approach is given in parentheses.

features used in PCEDNet [Himeur et al., 2021], which rely on point normals estimated using a small neighborhood of points, PCEDNet is by design not able to detect boundaries in point clouds. In contrast, using our proposed set of features and the extended *Default*++ dataset makes our approach capable of detecting boundaries in addition to sharp edges.

Figure 5.7 shows successfully detected boundaries for the two rightmost models, i.e. the only ones containing actual boundary points. For model 1222 of the *ABC* datasets evaluation data (see Figure 5.8), the boundary is found almost perfectly as well. Despite being actually 3D structures and therefore not boundaries in the strict sense, the top of the walls of model 0059 are detected as a boundary as well. Due to the low thickness of the walls, this is a reasonable behavior depending on the exact use-case for the extracted boundary data.

Very thin structures being identified correctly as boundary can also be seen in the red zoom-in of Figure 5.9. In the *station* point cloud (see Figure 5.10), mostly points of thin signs and humans are identified as boundary points. Note, that humans in this point cloud are mostly two-dimensional due to the scanning procedure and rather low resolution.

Finally, results on plants are depicted in Figure 5.11. All leaves are nicely separated by boundaries. Some stems contain sharp-edge points due to scanning artifacts.

5.5.7 Behavior on Noisy Data

In addition to the results on clean point clouds in Figure 5.8, Figure 5.12 shows a direct comparison on clean as well as noisy data taken from the *ABC* dataset of our algorithm BoundED and PCEDNet [Himeur et al., 2021]. The respective noisy models are taken from Himeur et al. [2021][8]. Note, that we assume only the point positions to be given. Thus, the normals needed by PCEDNet were calculated according to the authors instructions via meshlab [Cignoni et al., 2008]. BoundED outperforms PCEDNet on noisy data if both are trained on *Default* as it is significantly less prone to predict false positives in originally flat regions. The difference is particularly noticeable in the eighth row on model 7487. Furthermore, in the sixth row on model 4986, PCEDNet has difficulties in detecting the prominent sharp edges at the top and botom of the object. As the network architectures of both approaches are very similar, we expect that the main reason for our approach to perform better in the presence of noise is the additional robustness of our features due to the underlying statistics.

5.5.8 Ablation Study

In the scope of an additional ablation study, we validate the chosen features as well as the selected number of scales.

Table 5.4 shows median scores of various classification metrics for results of our approach trained and evaluated on the *Default*++ dataset. While all chosen features seem to contribute

[8] Available at: `https://storm-irit.github.io/pcednet-supp/abc_noise_0.04.html`, accessed on 10/19/2022.

	Precision(↑)	Recall(↑)	MCC(↑)	F1(↑)	Accuracy(↑)	IoU(↑)
$\boldsymbol{x}_{i,k} = (\boldsymbol{d}_{i,k}, \boldsymbol{s}_{i,k}, \boldsymbol{c}_{i,k})^T$	0.308	0.680	0.413	0.427	0.888	0.272
$\boldsymbol{x}_{i,k} = (\boldsymbol{\sigma}_{i,k})^T$	0.331	0.379	0.310	0.354	0.915	0.215
$\boldsymbol{x}_{i,k} = (\boldsymbol{\sigma}_{i,k,\text{upper}}, \boldsymbol{\sigma}_{i,k,\text{lower}})^T$	0.380	0.113	0.172	0.172	**0.934**	0.094
$\boldsymbol{x}_{i,k} = (\boldsymbol{\sigma}_{i,k,\text{upper}}, \boldsymbol{\sigma}_{i,k,\text{lower}}, \boldsymbol{s}_{i,k})^T$	0.361	0.743	0.464	0.474	0.903	0.311
$\boldsymbol{x}_{i,k} = (\boldsymbol{\sigma}_{i,k,\text{upper}}, \boldsymbol{\sigma}_{i,k,\text{lower}}, \boldsymbol{d}_{i,k}, \boldsymbol{s}_{i,k})^T$	0.423	0.711	0.499	0.518	0.923	0.349
$\boldsymbol{x}_{i,k} = (\boldsymbol{\sigma}_{i,k,\text{upper}}, \boldsymbol{\sigma}_{i,k,\text{lower}}, \boldsymbol{d}_{i,k}, \boldsymbol{c}_{i,k})^T$	0.416	0.248	0.296	0.326	0.933	0.195
$\boldsymbol{x}_{i,k} = (\boldsymbol{\sigma}_{i,k,\text{upper}}, \boldsymbol{\sigma}_{i,k,\text{lower}}, \boldsymbol{s}_{i,k}, \boldsymbol{c}_{i,k})^T$	0.327	0.760	0.449	0.454	0.889	0.293
$\boldsymbol{x}_{i,k} = (\boldsymbol{\sigma}_{i,k}, \boldsymbol{d}_{i,k}, \boldsymbol{s}_{i,k}, \boldsymbol{c}_{i,k})^T$	0.359	0.723	0.472	0.482	0.903	0.318
$\boldsymbol{x}_{i,k} = (\boldsymbol{\sigma}_{i,k,\text{upper}}, \boldsymbol{\sigma}_{i,k,\text{lower}}, \boldsymbol{d}_{i,k}, \boldsymbol{s}_{i,k}, \boldsymbol{c}_{i,k})^T$	**0.436**	0.709	0.522	**0.542**	0.927	**0.371**
$\boldsymbol{x}_{i,k} = (\boldsymbol{\sigma}_{i,k}, \boldsymbol{\sigma}_{i,k,\text{upper}}, \boldsymbol{\sigma}_{i,k,\text{lower}}, \boldsymbol{d}_{i,k}, \boldsymbol{s}_{i,k}, \boldsymbol{c}_{i,k})^T$	0.373	**0.866**	**0.529**	0.521	0.903	0.352

Table 5.4: Ablation study regarding the choice of features used as input for the network. The table lists median scores for various classification metrics. *Default*++ dataset is used for training as well as evaluation. The individual features are defined in Sections 5.4.1 and 5.4.2.

	Precision(↑)	Recall(↑)	MCC(↑)	F1(↑)	Accuracy(↑)	IoU(↑)
2 scales (128, 32)	0.313	**0.857**	0.470	0.456	0.876	0.295
4 scales (128, 64, 32, 16)	0.436	0.709	**0.522**	0.542	0.927	0.371
8 scales (128, 91, 64, 45, 32, 23, 16, 11)	**0.477**	0.639	0.515	0.542	**0.935**	0.372
16 scales (128, 108, 91, 76, 64, 54, 45, 38, 32, 27, 23, 19, 16, 13, 11, 10)	0.463	0.662	0.520	**0.545**	0.932	**0.375**

Table 5.5: Ablation study regarding the number of scales used by our network. The table lists median scores for various classification metrics. The *Default*++ dataset is used for training as well as evaluation.

positively to the classification result, experiments suggest, that $\boldsymbol{s}_{i,k}$ is the most important feature. We suspect the reason for this to be the high importance of its tangential component for the detection of boundaries while additionally the normal component can be utilized by the network to classify sharp-edge points. Also note, that the partitioning of the neighborhood according to the estimated normal degrades classification quality if the singular values of the respective covariance matrices are given to the network in isolation. However, if being combined with the other proposed features, the partioning improves the results. We reason, that the partitioning provides the network with additional cues for finetuning the results, but is too dependent on correct normal estimation to be suited for robust sharp-edge and boundary point detection without further information. Passing the singular values $\boldsymbol{\sigma}_{i,k}$ of the unpartitioned neighborhood's covariance matrix to the network in addition to $\boldsymbol{\sigma}_{i,k,\text{upper}}$ and $\boldsymbol{\sigma}_{i,k,\text{lower}}$ does not seem to improve the results significantly.

The impact of the number of scales is shown in Table 5.5. For the experiments, we chose to use 2^i neighbors per scale where i is distributed evenly-spaced over the interval $(3, 7]$. While the performance of our algorithm using 4, 8, or 16 scales is very similar, using 2 scales performs much worse. As a trade-off between performance and computational burden, we use four scales.

5.5.9 Limitations

Despite yielding great results in most cases, the feature extraction step can fail in various scenarios. If e.g. the smallest singular value of a neighborhood's covariance matrix does not correspond to the true surface normal, the points are partitioned in an unexpected way leading to very unpredictable results. Note, that, by estimating the normal per-scale and passing all respective singular values to the network, it is able to extract additional information about the neighborhood. Using only a single-scale normal e.g. estimated during the scanning process might therefore lead to less accurate classifications. To some degree, this can be compensated by the classification network given enough training data. Nonetheless, the results would surely improve if the feature extraction step can already tackle such edge cases on its own by e.g. using a per-scale global normal smoothing step.

Furthermore, the *Default* dataset was designed to yield good results if GLS features are used for classification, but it does not cover all relevant edge cases for our features. Outliers seem to be one such example. As they are heavily underrepresented in the training data, the classification network fails to distinguish those properly from points on the actual surface. Designing a new point cloud dataset with our features in mind or even generating a dataset based on the feature values directly instead of going the detour over generating point clouds could solve this problem.

Finally, depending on the point cloud size, a large part of the time needed for the feature extraction step is spent on finding the k neighbors of each point. A custom tailored solution for this neighborhood search could probably improve the performance of the feature extraction significantly. Due to the simplicity and compactness of the network, the same holds for the implementation of the classification network as general frameworks like PyTorch introduce a significant overhead in this situation.

5.6 Conclusion and Future Work

In this work we introduced a novel set of per-point features to facilitate the detection of sharp edges and boundaries via a simple and compact neural classification network. Due to the small network and an efficient GPU implementation for the feature extraction, the algorithm is faster than previous state-of-the-art methods while at the same time achieving more consistent classification results. This could make the proposed BoundED algorithm a good choice for situations in which interactive classification is required.

The two-level covariance analysis conducted on the neighborhood of a point has, even in the simple form deployed in this work, proven to be a valuable tool to describe the local geometry. In the future, our novel features could be utilized to estimate the curvature of curved surfaces as well. We expect the inclusion of higher order moments to further improve the results and enable us to also learn the estimation of distances to edges and boundaries.

Part III

Conclusion

Chapter 6

Conclusion

In the following, the contributions of the publications included in this thesis are summarized. Furthermore, limitations and opportunities for future work are discussed.

6.1 Contributions

As part of this thesis, we presented three different research projects in Chapters 3, 4, and 5 respectively. The goal of these projects was to develop techniques for improving the feasibility of state-of-the-art scene reconstruction solutions for applications relying on practical real-time reflectance estimation. In Chapter 1.1, we identified two main challenges that need to be addressed to achieve this goal: sparsity of available appearance samples and strong constraints regarding the processing time.

As appearance is the result of the complex interplay of illumination, geometry, as well as reflectance characteristics of the surface and additionally depends on the viewing direction, it can only be reconstructed accurately based on a dense sampling. For very smooth surfaces, the sampling has to be extremely dense, as some reflectance characteristics may only be observed from particular viewing directions. Unfortunately, sampling the appearance with this density is not feasible in many applications as it would negatively impact the user experience and increase hardware cost. It can even be entirely impossible due to physical constraints. Therefore, reflectance estimation algorithms have to be able to operate on sparse appearance samples, e.g., captured by a single webcam in a telepresence application or by a small number of sensors integrated into a head-mounted VR device.

Similarly, meeting certain time constraints is crucial for many applications as long processing times and high latency might hurt, e.g., social interaction in a telepresence setting or prevent an operator of a remotely controlled robot from reacting properly to a dynamic environment. Real-time online scene reconstruction, however, is challenging as vast amounts of data have to be processed, and resulting reconstructions have to be highly accurate as well as temporally stable. If this reconstruction is required to happen on a mobile device like a mobile phone or AR glasses with very limited computing power, it is even more critical to develop fast and efficient algorithms while maintaining the high visual quality already achieved by offline methods.

Real-time Reflectance Estimation from RGB-D In this project [Bode et al., 2019], we proposed a complete reflectance estimation solution focusing on meeting the time constraints by sacrificing some quality compared to offline approaches. Our method builds on a real-time geometry estimation pipeline and extends it with a reflectance estimation component. The main parts of this extension are an efficient GPU implementation for gathering and managing appearance samples over a given image sequence in real time, as well as the learning-based estimation of per-object Ward BRDF parameters for each frame and consecutive fusion over the sequence. Additionally, we proposed to perform a preliminary illumination estimation to enable our algorithm to reconstruct spatially-varying diffuse albedo, which captures further details. Utilizing a segmentation and the trained CNN prior allows the algorithm to yield reasonable reflectance estimates for the scene despite the sparsity of available appearance samples.

Locally-guided Neural Denoising In reflectance estimation pipelines, noise-like artifacts frequently cannot be avoided. Typically, the artifacts in the reconstructed data increase if the computational budget is very small or the appearance samples available are highly sparse, which is the case for many of the targeted applications. Instead of improving the reflectance estimation process, these artifacts can also be addressed in a post-processing step using image restoration algorithms like denoising methods. Most existing methods, however, are not suitable for the application to our typical estimated reflectance data as they assume homogeneous noise characteristics over the data. This assumption does not hold in our case, as the appearance sample sparsity problem is much more severe for very smooth surfaces than for rough, diffusely reflecting ones. In the second project presented as part of this thesis [Bode et al., 2022a], we proposed a novel technique for noise-level estimation and a scheme for using such information to guide state of the art learning-based image denoising methods with only minor modifications. We demonstrate that this approach is able to remove disturbing artifacts from natural images as well as reconstructed reflectance data while preserving fine details in initially clean parts.

Neural Boundary and Edge Detection Most existing pipelines for reflectance estimation leverage some kind of clustering or segmentation to overcome the sparsity of available appearance samples. While we used a geometry-based scene segmentation in our real-time reflectance estimation pipeline [Bode et al., 2019] resulting in a set of specular BRDF parameters per object, other options include clustering surface points according to their material or segmenting the scene into multiple segments per object according to, e.g., the individual parts it is assembled from. Crucial for calculating a meaningful segmentation is a good understanding of the scene, such as, e.g., information about the types of objects in the scene, the materials they consist of, their positions, or their extent. To this end, detecting edges and boundaries in the scene can benefit reflectance estimation pipelines. In the third project of this thesis [Bode et al., 2022b], we improved on the state-of-the-art in the field of edge and boundary detection in scenes represented as a point cloud, i.e., we assume only a position to be given for each surface point. We developed a small set of features that can describe local point neighborhoods well, such that a very compact MLP can classify points in non-edge, sharp-edge, or boundary. Due to the simplicity and compactness of the MLP, we expect our approach to be a good fit for real-time reflectance estimation pipelines.

6.2 Limitations and Outlook

Despite the tremendous progress in the field of real-time reflectance estimation over recent years, there is still a significant gap in the quality of the reconstructions between offline and online approaches. The reasons for this gap are manifold:

First, the reflectance models and light transport formulations commonly used in online pipelines are not expressive enough to adequately capture the variety of different appearance characteristics real-world scenes exhibit. Most existing approaches, e.g., do not handle complex materials involving transparency or subsurface scattering. Part of the problem is that the current generation of commodity depth sensors cannot handle transparent surfaces. Nonetheless, simpler models like the Phong BRDF [Phong, 1975] or Ward BRDF [Ward, 1992] are still popular as they are easier to be fitted to the sparse appearance data. More complex models like microfacet BRDFs are already quite common in offline reflectance estimation approaches, increasing the realism in reconstructed scenes. Even more expressive are neural reflectance models parameterized over neural features that are interpreted by a trained rendering network. It has been demonstrated that this neural shading technique can even correct for inaccuracies in the underlying geometry. Furthermore, neural radiance fields [Mildenhall et al., 2021] could be considered to represent the appearance in the reconstructed scene as they have been shown to excel at synthesizing photorealistic images from novel viewpoints. Currently, these techniques are too slow to be feasible for real-time online reflectance estimation pipelines, but the rapid progress in these research fields suggests that this might change in the near future.

Second, due to the strong time constraints coupled with the sparsity of the appearance data, the reflectance model fitting is not accurate enough to yield realistic reconstructions. For the offline reflectance estimation setting, this has been improved recently by leveraging differentiable rendering techniques, which are, however, not yet fast enough for integration into real-time pipelines at the moment. Outstanding results have been achieved, especially in cases where a good initialization is given. In the future, this property could be utilized to refine a reconstructed scene, with an initial reconstruction being obtained via classical online reflectance estimation pipelines. This way, the initial reconstruction is still available to the user with low latency, while the refined reconstruction can gradually replace it as the refined version becomes available. Additionally, some differentiable rendering techniques are able to handle global illumination effects like shadows or interreflections in the input data. Thus, additional information can be utilized to achieve more accurate estimates compared to most existing pipelines, which ignore these effects. Finally, differentiable rendering could also be used to refine the camera poses for the input images, as demonstrated in the offline setting already. Since online reconstruction pipelines are typically susceptible to camera tracking errors, this camera pose refinement could positively impact the overall reconstruction quality.

Third, many post-processing methods, like denoising the resulting estimates or the previously mentioned differentiable rendering-based refinement, are not yet fast enough for usage in an online reflectance estimation pipeline. Regarding denoising, this could potentially be solved by using (weakly-)supervised methods. While depending on a set of training data that has to fit the actual application, the denoising process tends to be faster than similar self-supervised methods.

For the future, we suggest coupling real-time reflectance estimation pipelines to scene understanding more tightly. This would allow the pipeline to adjust automatically to the degree of realism needed for a specific application. For a typical video conferencing setup, the face of a person is usually the most relevant part of the scene und should be reconstructed as accurately as possible. However, it would suffice only to match the appearance of the cloth the person is wearing on a macroscopic level instead of matching the exact underlying knitting or weaving pattern. A similar idea can often be applied to furniture in indoor settings: Instead of matching the exact texture of, e.g., a wooden chair, it might suffice just to apply a generic wood texture matching the type of wood that is used for the chair. Besides pooling of appearance samples, scene understanding could, hence, also be used to, e.g., query materials from a database for parts of the scene, which would allow the reconstruction pipeline to focus the limited computational resources on the essential parts of the scene according to the targeted application. Applying strong material-specific priors during the reflectance parameter fitting could also improve the robustness of the reconstruction pipeline and resulting realism.

Mostly orthogonal to improvements in the reconstruction algorithms, we also still see progress concerning the hardware. While consumer-grade photo and video cameras already yield almost noise-free images in typical settings, depth data typically exhibit much more noise. However, sophisticated sensors producing high-quality depth images are becoming more and more affordable, which is obviously beneficial for reconstruction pipelines. Furthermore, the computation hardware is becoming more powerful with each generation, and special-purpose hardware for efficient ray tracing and evaluation of neural networks is available in many modern GPUs. Finally, shifting computationally expensive parts of the reconstruction pipeline to the cloud could also improve results, especially on mobile or low-end devices.

While not achieving practical, high-quality real-time reflectance estimation just yet, we believe our contributions will positively impact the field and help to achieve the described goal in the near future.

Bibliography

Ahmed, Syeda Mariam, Yan Zhi Tan, Chee Meng Chew, Abdullah Al Mamun, and Fook Seng Wong (2018). "Edge and Corner Detection for Unorganized 3D Point Clouds with Application to Robotic Welding." *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Aittala, Miika, Tim Weyrich, Jaakko Lehtinen, et al. (2015). "Two-Shot SVBRDF Capture for Stationary Materials." *ACM Transactions on Graphics (TOG)*.

Albert, Rachel A, Dorian Yao Chan, Dan B Goldman, and James F O'Brien (2018). "Approximate svBRDF Estimation From Mobile Phone Video." *Eurographics Symposium on Rendering (EGSR)*.

Alexa, Marc, Johannes Behr, Daniel Cohen-Or, Shachar Fleishman, David Levin, and Claudio T Silva (2001). "Point Set Surfaces." *IEEE Visualization*.

Alliez, Pierre, Simon Giraudot, Clément Jamin, Florent Lafarge, Quentin Mérigot, Jocelyn Meyron, Laurent Saboret, Nader Salman, Shihao Wu, and Necip Fazil Yildiran (2022). "Point Set Processing." *CGAL User and Reference Manual*. 5.5. URL: `https://doc.cgal.org/5.5/Manual/packages.html%5Cchar%220023PkgPointSetProcessing3`.

Azinovic, Dejan, Tzu-Mao Li, Anton Kaplanyan, and Matthias Nießner (2019). "Inverse Path Tracing for Joint Material and Lighting Estimation." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Barron, Jonathan T and Jitendra Malik (2013). "Intrinsic Scene Properties from a Single RGB-D Image." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Barron, Jonathan T and Jitendra Malik (2014). "Shape, Illumination, and Reflectance from Shading." *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*.

Barron, Jonathan T, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan (2021). "Mip-NeRF: A Multiscale Representation for Anti-Aliasing Neural Radiance Fields." *IEEE International Conference on Computer Vision (ICCV)*.

Barron, Jonathan T., Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman (2022). "Mip-NeRF 360: Unbounded Anti-Aliased Neural Radiance Fields." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Barrow, Harry and Jay M Tenenbaum (1978). "Recovering intrinsic scene characteristics." *Computer Vision Systems*.

Bazazian, Dena, Josep R Casas, and Javier Ruiz-Hidalgo (2015). "Fast and Robust Edge Extraction in Unorganized Point Clouds." *International Conference on Digital Image Computing: Techniques and Applications (DICTA)*.

Bendels, Gerhard H, Ruwen Schnabel, and Reinhard Klein (2006). "Detecting Holes in Point Set Surfaces."

Bi, Sai, Nima Khademi Kalantari, and Ravi Ramamoorthi (2017). "Patch-Based Optimization for Image-Based Texture Mapping." *ACM Transactions on Graphics (TOG)*.

Blomley, Rosmarie and Martin Weinmann (2017). "USING MULTI-SCALE FEATURES FOR THE 3D SEMANTIC LABELING OF AIRBORNE LASER SCANNING DATA." *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*.

Bode, Lukas (2018). "Surface Material Recovery on RGB-D Image Sequences Using a CNN." *Master's Thesis at RWTH Aachen University*.

Bode, Lukas, Sebastian Merzbach, Julian Kaltheuner, Michael Weinmann, and Reinhard Klein (2022a). "Locally-guided Neural Denoising." *Graphics and Visual Computing (GVC)*. ISSN: 2666-6294. DOI: 10.1016/j.gvc.2022.200058. URL: https://www.sciencedirect.com/science/article/pii/S2666629422000110.

Bode, Lukas, Sebastian Merzbach, Patrick Stotko, Michael Weinmann, and Reinhard Klein (2019). "Real-time Multi-material Reflectance Reconstruction for Large-scale Scenes under Uncontrolled Illumination from RGB-D Image Sequences." *International Conference on 3D Vision (3DV)*. DOI: 10.1109/3DV.2019.00083.

Bode, Lukas, Michael Weinmann, and Reinhard Klein (2022b). "BoundED: Neural Boundary and Edge Detection in 3D Point Clouds via Local Neighborhood Statistics." *arXiv preprint arXiv:2210.13305, submitted to ISPRS Journal of Photogrammetry and Remote Sensing (P&RS) (under review)*. DOI: 10.48550/arXiv.2210.13305.

Bonneel, Nicolas, Balazs Kovacs, Sylvain Paris, and Kavita Bala (2017). "Intrinsic Decompositions for Image Editing." *Computer Graphics Forum (CGF)*.

Boulch, Alexandre, Bertrand Le Saux, and Nicolas Audebert (2017). "Unstructured Point Cloud Semantic Labeling Using Deep Segmentation Networks." *Eurographics Workshop on 3D Object Retrieval (3DOR)*. DOI: 10.2312/3dor.20171047.

Brodu, Nicolas and Dimitri Lague (2012). "3D terrestrial lidar data classification of complex natural scenes using a multi-scale dimensionality criterion: Applications in geomorphology." *ISPRS Journal of Photogrammetry and Remote Sensing (P&RS)*.

Buades, Antoni, Bartomeu Coll, and Jean Michel Morel (2005). "A non-local algorithm for image denoising." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Chaitanya, Chakravarty R Alla, Anton S Kaplanyan, Christoph Schied, Marco Salvi, Aaron Lefohn, Derek Nowrouzezahrai, and Timo Aila (2017). "Interactive Reconstruction of Monte Carlo Image Sequences using a Recurrent Denoising Autoencoder." *ACM Transactions on Graphics (TOG)*.

Chambolle, Antonin (2004). "An Algorithm for Total Variation Minimization and Applications." *Journal of Mathematical Imaging and Vision*.

Che, Erzhuo and Michael J Olsen (2018). "Multi-scan segmentation of terrestrial laser scanning data based on normal variation analysis." *ISPRS Journal of Photogrammetry and Remote Sensing (P&RS)*.

Chen, Hui, Man Liang, Wanquan Liu, Weina Wang, and Peter Xiaoping Liu (2022a). "An Approach to Boundary Detection for 3D Point Clouds Based on DBSCAN Clustering." *Pattern Recognition*. ISSN: 0031-3203. DOI: `10.1016/j.patcog.2021.108431`. URL: `https://www.sciencedirect.com/science/article/pii/S0031320321006075`.

Chen, Jiawen, Dennis Bautembach, and Shahram Izadi (2013a). "Scalable Real-time Volumetric Surface Reconstruction." *ACM Transactions on Graphics (TOG)*.

Chen, Jiazhou, Gaël Guennebaud, Pascal Barla, and Xavier Granier (2013b). "Non-Oriented MLS Gradient Fields." *Computer Graphics Forum (CGF)*.

Chen, Mingqin, Yuhui Quan, Tongyao Pang, and Hui Ji (2022b). "Nonblind Image Deconvolution via Leveraging Model Uncertainty in An Untrained Deep Neural Network." *International Journal of Computer Vision (IJCV)*.

Chen, Qifeng and Vladlen Koltun (2013). "A Simple Model for Intrinsic Image Decomposition with Depth Cues." *IEEE International Conference on Computer Vision (ICCV)*.

Chen, Yun-Chun, Chen Gao, Esther Robb, and Jia-Bin Huang (2020). "NAS-DIP: Learning Deep Image Prior with Neural Architecture Search."

Cheng, Zezhou, Matheus Gadelha, Subhransu Maji, and Daniel Sheldon (2019a). "A Bayesian Perspective on the Deep Image Prior." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Cheng, Ziang, Yinqiang Zheng, Shaodi You, and Imari Sato (2019b). "Non-local Intrinsic Decomposition with Near-infrared Priors." *IEEE International Conference on Computer Vision (ICCV)*.

Cignoni, Paolo, Marco Callieri, Massimiliano Corsini, Matteo Dellepiane, Fabio Ganovelli, and Guido Ranzuglia (2008). "MeshLab: an Open-Source Mesh Processing Tool." *Eurographics Italian Chapter Conference*. Edited by Vittorio Scarano, Rosario De Chiara, and Ugo Erra. DOI: `10.2312/LocalChapterEvents/ItalChap/ItalianChapConf2008/129-136`.

Conn, Adam, Ullas V Pedmale, Joanne Chory, Charles F Stevens, and Saket Navlakha (2017). "A Statistical Description of Plant Shoot Architecture." *Current Biology*.

Dabov, Kostadin, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian (2007a). "Color image denoising via sparse 3D collaborative filtering with grouping constraint in luminance-chrominance space." *IEEE International Conference on Image Processing*.

Dabov, Kostadin, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian (2007b). "Image Denoising by Sparse 3-D Transform-Domain Collaborative Filtering." *IEEE Transactions on Image Processing*.

Dai, Angela, Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Christian Theobalt (2017). "BundleFusion: Real-Time Globally Consistent 3D Reconstruction Using On-the-Fly Surface Reintegration." *ACM Transactions on Graphics (TOG)*.

Dai, Angela, Yawar Siddiqui, Justus Thies, Julien Valentin, and Matthias Nießner (2021). "SPSG: Self-Supervised Photometric Scene Generation from RGB-D Scans." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Daniels Ii, Joel, Tilo Ochotta, Linh K Ha, and Cláudio T Silva (2008). "Spline-based feature curves from point-sampled geometry." *The Visual Computer*.

Demantké, Jérôme, Clément Mallet, Nicolas David, and Bruno Vallet (2011). "Dimensionality based scale selection in 3D lidar point clouds." *Laserscanning*.

Demarsin, Kris, Denis Vanderstraeten, Tim Volodine, and Dirk Roose (2007). "Detection of closed sharp edges in point clouds using normal estimation and graph theory." *Computer Aided Design*.

Diaz, Mauricio and Peter Sturm (2013). "Estimating Photometric Properties from Image Collections." *Journal of Mathematical Imaging and Vision*.

Dong, Yue, Guojun Chen, Pieter Peers, Jiawan Zhang, and Xin Tong (2014). "Appearance-from-Motion: Recovering Spatially Varying Surface Reflectance under Unknown Lighting." *ACM Transactions on Graphics (TOG)*.

Edelsbrunner, Herbert and Ernst P Mücke (1994). “Three-dimensional alpha shapes.” *ACM Transactions on Graphics (TOG)*.

Elad, Michael and Michal Aharon (2006). “Image Denoising Via Sparse and Redundant Representations Over Learned Dictionaries.” *IEEE Transactions on Image Processing*.

Epic Games (2022). *Unreal Engine*. URL: `https://www.unrealengine.com`.

Fan, Hangming, Rui Wang, Yuchi Huo, and Hujun Bao (2021). “Real-time Monte Carlo Denoising with Weight Sharing Kernel Prediction Network.” *Computer Graphics Forum (CGF)*.

Fan, Ting-Jun, Gerard Medioni, and Ramakant Nevatia (1987). “Segmented descriptions of 3-D surfaces.” *IEEE International Conference on Robotics and Automation (ICRA)*.

Fernandes, Leandro AF and Manuel M Oliveira (2012). “A General Framework for Subspace Detection in Unordered Multidimensional Data.” *Pattern Recognition*.

Fleishman, Shachar, Daniel Cohen-Or, and Cláudio T Silva (2005). “Robust Moving Least-squares Fitting with Sharp Features.” *ACM Transactions on Graphics (TOG)*.

Foi, Alessandro, Vladimir Katkovnik, and Karen Egiazarian (2006). “Pointwise shape-adaptive DCT denoising with structure preservation in luminance-chrominance space.” *International Workshop on Video Processing and Quality Metrics for Consumer Electronics (VPQM)*.

Fu, Yanping, Qingan Yan, Jie Liao, Huajian Zhou, Jin Tang, and Chunxia Xiao (2021). “Seamless Texture Optimization for RGB-D Reconstruction.” *IEEE Transactions on Visualization and Computer Graphics (TVCG)*.

Gadelha, Matheus, Rui Wang, and Subhransu Maji (2019). “Shape Reconstruction using Differentiable Projections and Deep Priors.” *IEEE International Conference on Computer Vision (ICCV)*.

Garces, Elena, Carlos Rodriguez-Pardo, Dan Casas, and Jorge Lopez-Moreno (2022). “A Survey on Intrinsic Images: Delving Deep Into Lambert and Beyond.” *International Journal of Computer Vision (IJCV)*.

Geisler-Moroder, David and Arne Dür (2010). “A New Ward BRDF Model with Bounded Albedo.” *Computer Graphics Forum (CGF)*.

Gelfand, Natasha and Leonidas J Guibas (2004). “Shape segmentation using local slippage analysis.” *Symposium on Geometry Processing (SGP)*.

Georgoulis, Stamatios, Konstantinos Rematas, Tobias Ritschel, Mario Fritz, Luc Van Gool, and Tinne Tuytelaars (2016). "DeLight-Net: Decomposing Reflectance Maps into Specular Materials and Natural Illumination." *arXiv preprint arXiv:1603.08240*.

Goldman, Dan B, Brian Curless, Aaron Hertzmann, and Steven M Seitz (2009). "Shape and Spatially-Varying BRDFs From Photometric Stereo." *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*.

Gretton, Arthur, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola (2012). "A Kernel Two-Sample Test." *Journal of Machine Learning Research (JMLR)*.

Guennebaud, Gaël and Markus Gross (2007). "Algebraic Point Set Surfaces." *Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*.

Guerrero, Paul, Yanir Kleiman, Maks Ovsjanikov, and Niloy J Mitra (2018). "PCPNet: Learning Local Shape Properties from Raw Point Clouds." *Computer Graphics Forum (CGF)*.

Gumhold, Stefan, Xinlong Wang, and Rob S MacLeod (2001). "Feature Extraction From Point Clouds." *International Meshing Roundtable (IMR)*.

Guo, Bao, Yuhe Zhang, Jian Gao, Chunhui Li, and Yao Hu (2022). "SGLBP: Subgraph-based Local Binary Patterns for Feature Extraction on Point Clouds." *Computer Graphics Forum (CGF)*.

Guo, Yulan, Hanyun Wang, Qingyong Hu, Hao Liu, Li Liu, and Mohammed Bennamoun (2020). "Deep Learning for 3D Point Clouds: A Survey." *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*.

Haber, Tom, Christian Fuchs, Philippe Bekaer, Hans-Peter Seidel, Michael Goesele, and Hendrik PA Lensch (2009). "Relighting Objects from Image Collections." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Hachama, Mohammed, Bernard Ghanem, and Peter Wonka (2015). "Intrinsic Scene Decomposition from RGB-D images." *IEEE International Conference on Computer Vision (ICCV)*.

Hackel, Timo, Nikolay Savinov, Lubor Ladicky, Jan D. Wegner, Konrad Schindler, and Marc Pollefeys (2017). "SEMANTIC3D.NET: A new large-scale point cloud classification benchmark." *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*.

Hackel, Timo, Jan D Wegner, and Konrad Schindler (2016a). "Contour detection in unstructured 3D point clouds." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Hackel, Timo, Jan D Wegner, and Konrad Schindler (2016b). “Fast semantic segmentation of 3D point clouds with strongly varying density.” *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*.

Hanocka, Rana, Gal Metzer, Raja Giryes, and Daniel Cohen-Or (2020). “Point2Mesh: A Self-Prior for Deformable Meshes.”

Hart, John C, Daniel J Sandin, and Louis H Kauffman (1989). “Ray tracing deterministic 3-D fractals.” *Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*.

Heckel, Reinhard and Paul Hand (2019). “Deep Decoder: Concise Image Representations from Untrained Non-convolutional Networks.” *International Conference on Learning Representations (ICLR)*.

Himeur, Chems-Eddine, Thibault Lejemble, Thomas Pellegrini, Mathias Paulin, Loic Barthe, and Nicolas Mellado (2021). “PCEDNet: A Lightweight Neural Network for Fast and Interactive Edge Detection in 3D Point Clouds.” *ACM Transactions on Graphics (TOG)*.

Ho, Kary, Andrew Gilbert, Hailin Jin, and John Collomosse (2021). “Neural Architecture Search for Deep Image Prior.” *Computers & Graphics*.

Hu, Zeyu, Mingmin Zhen, Xuyang Bai, Hongbo Fu, and Chiew-lan Tai (2020). “JSENet: Joint Semantic Segmentation and Edge Detection Network for 3D Point Clouds.” *European Conference on Computer Vision (ECCV)*.

Huang, Jing and Suya You (2016). “Point Cloud Labeling using 3D Convolutional Neural Network.” *International Conference on Pattern Recognition (ICPR)*.

Huang, Jingwei, Justus Thies, Angela Dai, Abhijit Kundu, Chiyu Jiang, Leonidas J Guibas, Matthias Nießner, Thomas Funkhouser, et al. (2020). “Adversarial Texture Optimization from RGB-D Scans.” *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Izadi, Shahram, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, et al. (2011). “KinectFusion: Real-time 3D Reconstruction and Interaction Using a Moving Depth Camera.” *ACM Symposium on User Interface Software and Technology (UIST)*.

Kähler, Olaf, Victor Prisacariu, Julien Valentin, and David Murray (2015a). “Hierarchical Voxel Block Hashing for Efficient Integration of Depth Images.” *IEEE Robotics and Automation Letters*.

Kähler, Olaf, Victor A Prisacariu, and David W Murray (2016). “Real-Time Large-Scale Dense 3D Reconstruction with Loop Closure.” *European Conference on Computer Vision (ECCV)*.

Kähler, Olaf, Victor Adrian Prisacariu, Carl Yuheng Ren, Xin Sun, Philip Torr, and David Murray (2015b). "Very High Frame Rate Volumetric Integration of Depth Images on Mobile Devices." *IEEE Transactions on Visualization and Computer Graphics (TVCG)*.

Kajiya, James T (1986). "The rendering equation." *Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*.

Kaltheuner, Julian, Lukas Bode, and Reinhard Klein (2021). "Capturing Anisotropic SVBRDFs." *International Symposium on Vision, Modeling, and Visualization (VMV)*. DOI: `10.2312/vmv.20211372`.

Kattamis, Andreas, Tameem Adel, and Adrian Weller (2019). "Exploring Properties of the Deep Image Prior." *Advances in Neural Information Processing Systems (NeurIPS)*.

Kerl, Christian, Mohamed Souiai, Jürgen Sturm, and Daniel Cremers (2014). "Towards Illumination-invariant 3D Reconstruction using ToF RGB-D Cameras." *International Conference on 3D Vision (3DV)*.

Kim, Jungeon, Hyomin Kim, Hyeonseo Nam, Jaesik Park, and Seungyong Lee (2022). "TextureMe: High-Quality Textured Scene Reconstruction in Real Time." *ACM Transactions on Graphics (TOG)*.

Kim, Kihwan, Jinwei Gu, Stephen Tyree, Pavlo Molchanov, Matthias Nießner, and Jan Kautz (2017). "A Lightweight Approach for On-the-Fly Reflectance Estimation." *IEEE International Conference on Computer Vision (ICCV)*.

Kingma, Diederik P and Jimmy Ba (2014). "Adam: A Method for Stochastic Optimization." *arXiv preprint arXiv:1412.6980*.

Knecht, Martin, Georg Tanzmeister, Christoph Traxler, and Michael Wimmer (2012). "Interactive BRDF Estimation for Mixed-Reality Applications."

Koch, Sebastian, Albert Matveev, Zhongshi Jiang, Francis Williams, Alexey Artemov, Evgeny Burnaev, Marc Alexa, Denis Zorin, and Daniele Panozzo (2019). "ABC: A Big CAD Model Dataset For Geometric Deep Learning." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Laine, Samuli, Janne Hellsten, Tero Karras, Yeongho Seol, Jaakko Lehtinen, and Timo Aila (2020). "Modular Primitives for High-Performance Differentiable Rendering." *ACM Transactions on Graphics (TOG)*.

Landrieu, Loic, Hugo Raguet, Bruno Vallet, Clément Mallet, and Martin Weinmann (2017). "A structured regularization framework for spatially smoothing semantic labelings of 3D point clouds." *ISPRS Journal of Photogrammetry and Remote Sensing (P&RS)*.

Landrieu, Loic and Martin Simonovsky (2018). "Large-scale Point Cloud Semantic Segmentation with Superpoint Graphs." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Lassner, Christoph and Michael Zollhofer (2021). "Pulsar: Efficient Sphere-based Neural Rendering." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Lawin, Felix Järemo, Martin Danelljan, Patrik Tosteberg, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg (2017). "Deep Projective 3D Semantic Segmentation." *International Conference on Computer Analysis of Images and Patterns (CAIP)*.

Lee, Joo Ho, Hyunho Ha, Yue Dong, Xin Tong, and Min H Kim (2020). "TextureFusion: High-Quality Texture Acquisition for Real-Time RGB-D Scanning." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Lee, Lik-Hang, Tristan Braud, Pengyuan Zhou, Lin Wang, Dianlei Xu, Zijun Lin, Abhishek Kumar, Carlos Bermejo, and Pan Hui (2021). "All one needs to know about metaverse: A complete survey on technological singularity, virtual ecosystem, and research agenda." *arXiv preprint arXiv:2110.05352*.

Li, Bao, Ruwen Schnabel, Reinhard Klein, Zhiquan Cheng, Gang Dang, and Shiyao Jin (2010). "Robust normal estimation for point clouds with sharp features." *Computers & Graphics*.

Li, Jihao, Martin Weinmann, Xian Sun, Wenhui Diao, Yingchao Feng, Stefan Hinz, and Kun Fu (2022). "VD-LAB: A view-decoupled network with local-global aggregation bridge for airborne laser scanning point cloud classification." *ISPRS Journal of Photogrammetry and Remote Sensing (P&RS)*.

Li, Mingyu and Koichi Hashimoto (2017). "Curve Set Feature-Based Robust and Fast Pose Estimation Algorithm." *Sensors*.

Li, Shuda, Ankur Handa, Yang Zhang, and Andrew Calway (2016). "HDRFusion: HDR SLAM using a low-cost auto-exposure RGB-D sensor." *International Conference on 3D Vision (3DV)*.

Li, Tzu-Mao, Miika Aittala, Frédo Durand, and Jaakko Lehtinen (2018). "Differentiable Monte Carlo Ray Tracing through Edge Sampling." *ACM Transactions on Graphics (TOG)*.

Li, Xiao, Yue Dong, Pieter Peers, and Xin Tong (2017). "Modeling Surface Appearance from a Single Photograph using Self-augmented Convolutional Neural Networks." *ACM Transactions on Graphics (TOG)*.

Li, Zhengqin, Mohammad Shafiei, Ravi Ramamoorthi, Kalyan Sunkavalli, and Manmohan Chandraker (2020). “Inverse Rendering for Complex Indoor Scenes: Shape, Spatially-Varying Lighting and SVBRDF from a Single Image.” *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Lin, Tsung-Yi, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár (2017). “Focal Loss for Dense Object Detection.” *IEEE International Conference on Computer Vision (ICCV)*.

Lin, Yangbin, Cheng Wang, Jun Cheng, Bili Chen, Fukai Jia, Zhonggui Chen, and Jonathan Li (2015). “Line segment extraction for large scale unorganized point clouds.” *ISPRS Journal of Photogrammetry and Remote Sensing (P&RS)*.

Liu, Guilin, Duygu Ceylan, Ersin Yumer, Jimei Yang, and Jyh-Ming Lien (2017). “Material Editing Using a Physically Based Rendering Network.” *IEEE International Conference on Computer Vision (ICCV)*.

Liu, Jiaming, Yu Sun, Xiaojian Xu, and Ulugbek S Kamilov (2019). “Image Restoration using Total Variation Regularized Deep Image Prior.” *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Loizou, Marios, Melinos Averkiou, and Evangelos Kalogerakis (2020). “Learning Part Boundaries from 3D Point Clouds.” *Computer Graphics Forum (CGF)*.

Lombardi, Stephen and Ko Nishino (2012a). “Reflectance and Natural Illumination from a Single Image.” *European Conference on Computer Vision (ECCV)*.

Lombardi, Stephen and Ko Nishino (2012b). “Single Image Multimaterial Estimation.” *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Lombardi, Stephen and Ko Nishino (2015). “Reflectance and Illumination Recovery in the Wild.” *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*.

Lombardi, Stephen and Ko Nishino (2016). “Radiometric Scene Decomposition: Scene Reflectance, Illumination, and Geometry from RGB-D Images.” *International Conference on 3D Vision (3DV)*.

Lorensen, William E and Harvey E Cline (1987). “Marching cubes: A high resolution 3D surface construction algorithm.” *Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*.

Lu, Xiaohu, Yahui Liu, and Kai Li (2019). “Fast 3D Line Segment Detection From Unorganized Point Cloud.” *arXiv preprint arXiv:1901.02532*.

Maier, Robert, Kihwan Kim, Daniel Cremers, Jan Kautz, and Matthias Nießner (2017). "Intrinsic3D: High-Quality 3D Reconstruction by Joint Appearance and Geometry Optimization with Spatially-Varying Lighting." *IEEE International Conference on Computer Vision (ICCV)*.

Mairal, Julien, Michael Elad, and Guillermo Sapiro (2007). "Sparse Representation for Color Image Restoration." *IEEE Transactions on Image Processing*.

Mao, Yongqiang, Kaiqiang Chen, Wenhui Diao, Xian Sun, Xiaonan Lu, Kun Fu, and Martin Weinmann (2022). "Beyond single receptive field: A receptive field fusion-and-stratification network for airborne laser scanning point cloud classification." *ISPRS Journal of Photogrammetry and Remote Sensing (P&RS)*.

Mataev, Gary, Peyman Milanfar, and Michael Elad (2019). "DeepRED: Deep Image Prior Powered by RED." *IEEE International Conference on Computer Vision Workshops*.

Matveev, Albert, Ruslan Rakhimov, Alexey Artemov, Gleb Bobrovskikh, Vage Egiazarian, Emil Bogomolov, Daniele Panozzo, Denis Zorin, and Evgeny Burnaev (2022). "DEF: Deep Estimation of Sharp Geometric Features in 3D Shapes." *ACM Transactions on Graphics (TOG)*.

Maximov, Maxim, Laura Leal-Taixé, Mario Fritz, and Tobias Ritschel (2019). "Deep Appearance Maps." *IEEE International Conference on Computer Vision (ICCV)*.

Meka, Abhimitra, Gereon Fox, Michael Zollhöfer, Christian Richardt, and Christian Theobalt (2017). "Live User-Guided Intrinsic Video for Static Scenes." *IEEE Transactions on Visualization and Computer Graphics (TVCG)*.

Meka, Abhimitra, Maxim Maximov, Michael Zollhoefer, Avishek Chatterjee, Hans-Peter Seidel, Christian Richardt, and Christian Theobalt (2018). "LIME: Live Intrinsic Material Estimation." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Meka, Abhimitra, Mohammad Shafiei, Michael Zollhöfer, Christian Richardt, and Christian Theobalt (2021). "Real-time Global Illumination Decomposition of Videos." *ACM Transactions on Graphics (TOG)*.

Meka, Abhimitra, Michael Zollhöfer, Christian Richardt, and Christian Theobalt (2016). "Live Intrinsic Video." *ACM Transactions on Graphics (TOG)*.

Mellado, Nicolas, Matteo Dellepiane, and Roberto Scopigno (2015). "Relative Scale Estimation and 3D Registration of Multi-Modal Geometry Using Growing Least Squares." *IEEE Transactions on Visualization and Computer Graphics (TVCG)*. ISSN: 1077-2626. DOI: `10.1109/TVCG.2015.2505287`.

Mellado, Nicolas, Gaël Guennebaud, Pascal Barla, Patrick Reuter, and Christophe Schlick (2012). "Growing Least Squares for the Analysis of Manifolds in Scale-Space." *Computer Graphics Forum (CGF)*.

Mérigot, Quentin, Maks Ovsjanikov, and Leonidas J Guibas (2011). "Voronoi-Based Curvature and Feature Estimation from Point Clouds." *IEEE Transactions on Visualization and Computer Graphics (TVCG)*.

Merzbach, Sebastian, Max Hermann, Martin Rump, and Reinhard Klein (2019). "Learned Fitting of Spatially Varying BRDFs." *Computer Graphics Forum (CGF)*.

Mildenhall, Ben, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng (2021). "NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis." *Communications of the ACM*.

Mineo, Carmelo, Stephen Gareth Pierce, and Rahul Summan (2019). "Novel algorithms for 3D surface point cloud boundary detection and edge reconstruction." *Journal of Computational Design and Engineering*.

Mitropoulou, Aikaterini and Andreas Georgopoulos (2019). "AN AUTOMATED PROCESS TO DETECT EDGES IN UNORGANIZED POINT CLOUDS." *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*.

Miyata, Takamichi (2015). "Inter-channel relation based vectorial total variation for color image recovery." *IEEE International Conference on Image Processing*.

Monga, Olivier, Rachid Deriche, and Jean-Marie Rocchisani (1991). "3D edge detection using recursive filtering: Application to scanner images." *CVGIP: Image Understanding*.

Müller, Jan U, Michael Weinmann, and Reinhard Klein (2022a). "Unbiased Gradient Estimation for Differentiable Surface Splatting via Poisson Sampling." *European Conference on Computer Vision (ECCV)*.

Müller, Thomas, Alex Evans, Christoph Schied, and Alexander Keller (2022b). "Instant Neural Graphics Primitives with a Multiresolution Hash Encoding." *arXiv preprint arXiv:2201.05989*.

Neshatavar, Reyhaneh, Mohsen Yavartanoo, Sanghyun Son, and Kyoung Mu Lee (2022). "CVF-SID: Cyclic multi-Variate Function for Self-Supervised Image Denoising by Disentangling Noise from Image." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Newcombe, Richard A., Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J. Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon (2011). "KinectFusion: Real-Time Dense Surface Mapping and Tracking." *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*.

Nguyen, Keith Wei Liang, A Aprilia, Ahmad Khairyanto, Wee Ching Pang, Gerald Gim Lee Seet, and Shu Beng Tor (2018). “Edge detection from point cloud of worn parts.” *International Conference on Progress in Additive Manufacturing (Pro-AM)*.

Nguyen, Van Sinh, Trong Hai Trinh, and Manh Ha Tran (2015). “Hole Boundary Detection of a Surface of 3D Point Clouds.” *International Conference on Advanced Computing and Applications (ACOMP)*.

Nguyen-Phuoc, Thu H, Chuan Li, Stephen Balaban, and Yongliang Yang (2018). “RenderNet: A deep convolutional network for differentiable rendering from 3D shapes.” *Advances in Neural Information Processing Systems (NeurIPS)*.

Ni, Huan, Xiangguo Lin, Xiaogang Ning, and Jixian Zhang (2016). “Edge Detection and Feature Line Tracing in 3D-Point Clouds by Analyzing Geometric Properties of Neighborhoods.” *Remote Sensing*.

Niemeyer, Joachim, Franz Rottensteiner, and Uwe Soergel (2014). “Contextual classification of lidar data and building object detection in urban areas.” *ISPRS Journal of Photogrammetry and Remote Sensing (P&RS)*.

Nießner, Matthias, Michael Zollhöfer, Shahram Izadi, and Marc Stamminger (2013). “Real-time 3D Reconstruction at Scale using Voxel Hashing.” *ACM Transactions on Graphics (TOG)*.

Nimier-David, Merlin, Delio Vicini, Tizian Zeltner, and Wenzel Jakob (2019). “Mitsuba 2: A Retargetable Forward and Inverse Renderer.” *ACM Transactions on Graphics (TOG)*.

Oechsle, Michael, Lars Mescheder, Michael Niemeyer, Thilo Strauss, and Andreas Geiger (2019). “Texture Fields: Learning Texture Representations in Function Space.” *IEEE International Conference on Computer Vision (ICCV)*.

Osher, Stanley, Martin Burger, Donald Goldfarb, Jinjun Xu, and Wotao Yin (2005). “An iterative regularization method for total variation-based image restoration.” *Multiscale Modeling & Simulation*.

Öztireli, Cengiz, Gaël Guennebaud, and Markus Gross (2009). “Feature Preserving Point Set Surfaces based on Non-Linear Kernel Regression.” *Computer Graphics Forum (CGF)*.

Palma, Gianpaolo, Marco Callieri, Matteo Dellepiane, and Roberto Scopigno (2012). “A Statistical Method for SVBRDF Approximation from Video Sequences in General Lighting Conditions.” *Computer Graphics Forum (CGF)*.

Pang, Tongyao, Huan Zheng, Yuhui Quan, and Hui Ji (2021). “Recorrupted-to-Recorrupted: Unsupervised Deep Learning for Image Denoising.” *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Parker, Steven G, James Bigler, Andreas Dietrich, Heiko Friedrich, Jared Hoberock, David Luebke, David McAllister, Morgan McGuire, Keith Morley, Austin Robison, et al. (2010). "OptiX: A General Purpose Ray Tracing Engine." *ACM Transactions on Graphics (TOG)*.

Paszke, Adam, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala (2019). "PyTorch: An Imperative Style, High-Performance Deep Learning Library." *Advances in Neural Information Processing Systems (NeurIPS)*. URL: `http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf`.

Pauly, Mark, Richard Keiser, and Markus Gross (2003). "Multi-scale Feature Extraction on Point-Sampled Surfaces." *Computer Graphics Forum (CGF)*.

Pellacini, Fabio, James A Ferwerda, and Donald P Greenberg (2000). "Toward a Psychophysically-Based Light Reflection Model for Image Synthesis." *Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*.

Phong, Bui Tuong (1975). "Illumination for computer generated pictures." *Communications of the ACM*.

Pumarola, Albert, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer (2021). "D-NeRF: Neural Radiance Fields for Dynamic Scenes." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Qi, Charles R, Hao Su, Kaichun Mo, and Leonidas J Guibas (2017a). "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Qi, Charles Ruizhongtai, Li Yi, Hao Su, and Leonidas J Guibas (2017b). "PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space." *Advances in Neural Information Processing Systems (NeurIPS)*. URL: `https://proceedings.neurips.cc/paper/2017/file/d8bf84be3800d12f74d8b05e9b89836f-Paper.pdf`.

Quan, Yuhui, Mingqin Chen, Tongyao Pang, and Hui Ji (2020). "Self2Self With Dropout: Learning Self-Supervised Denoising From Single Image." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Quan, Yuhui, Yixin Chen, Yizhen Shao, Huan Teng, Yong Xu, and Hui Ji (2021). "Image Denoising Using Complex-Valued Deep CNN." *Pattern Recognition*.

Raina, Prashant, Sudhir Mudur, and Tiberiu Popa (2019). "Sharpness fields in point clouds using deep learning." *Computers & Graphics*. ISSN: 0097-8493. DOI: 10.1016/j.cag.2018.11.003. URL: https://www.sciencedirect.com/science/article/pii/S009784931830181X.

Raina, Prashant, Sudhir P Mudur, and Tiberiu Popa (2018). "MLS2: Sharpness Field Extraction Using CNN for Surface Reconstruction." *Graphics Interface (GI)*.

Rajwade, Ajit, Anand Rangarajan, and Arunava Banerjee (2012). "Image Denoising using the Higher Order Singular Value Decomposition." *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*.

Ravi, Nikhila, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari (2020). "Accelerating 3D Deep Learning with PyTorch3D." *arXiv:2007.08501*.

Rematas, Konstantinos, Tobias Ritschel, Mario Fritz, Efstratios Gavves, and Tinne Tuytelaars (2016). "Deep Reflectance Maps." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Richter-Trummer, Thomas, Denis Kalkofen, Jinwoo Park, and Dieter Schmalstieg (2016). "Instant Mixed Reality Lighting from Casual Scanning." *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*.

Romeiro, Fabiano and Todd Zickler (2010). "Blind Reflectometry." *European Conference on Computer Vision (ECCV)*.

Ronneberger, Olaf, Philipp Fischer, and Thomas Brox (2015). "U-Net: Convolutional Networks for Biomedical Image Segmentation." *International Conference on Medical Image Computing and Computer-assisted Intervention*.

Ruiters, Roland and Reinhard Klein (2009). "Heightfield and spatially varying BRDF Reconstruction for Materials with Interreflections." *Computer Graphics Forum (CGF)*.

Ruiters, Roland, Christopher Schwartz, and Reinhard Klein (2012). "Data Driven Surface Reflectance from Sparse and Irregular Samples." *Computer Graphics Forum (CGF)*.

Rusu, Radu Bogdan, Zoltan Csaba Marton, Nico Blodow, Mihai Dolha, and Michael Beetz (2008). "Towards 3D Point cloud based object maps for household environments." *Robotics and Autonomous Systems*.

Shi, Jian, Yue Dong, Xin Tong, and Yanyun Chen (2015). "Efficient intrinsic image decomposition for RGBD images." *ACM Symposium on User Interface Software and Technology (UIST)*.

Sidorov, Oleksii and Jon Yngve Hardeberg (2019). "Deep Hyperspectral Prior: Single-Image Denoising, Inpainting, Super-Resolution." *IEEE International Conference on Computer Vision Workshops*.

Srivastava, Nitish, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov (2014). "Dropout: A Simple Way to Prevent Neural Networks from Overfitting." *Journal of Machine Learning Research (JMLR)*.

Steinsiek, Moritz, Przemyslaw Polewski, Wei Yao, and Peter Krzystek (2017). "Semantische Analyse von ALS- und MLS-Daten in urbanen Gebieten mittels Conditional Random Fields." *Wissenschaftlich-Technische Jahrestagung der DGPF*.

Stotko, Patrick, Stefan Krumpen, Matthias B Hullin, Michael Weinmann, and Reinhard Klein (2019a). "SLAMCast: Large-Scale, Real-Time 3D Reconstruction and Streaming for Immersive Multi-Client Live Telepresence." *IEEE Transactions on Visualization and Computer Graphics (TVCG)*.

Stotko, Patrick, Michael Weinmann, and Reinhard Klein (2019b). "Albedo estimation for real-time 3D reconstruction using RGB-D and IR data." *ISPRS Journal of Photogrammetry and Remote Sensing (P&RS)*.

Tabib, Ramesh Ashok, Yashaswini V Jadhav, Swathi Tegginkeri, Kiran Gani, Chaitra Desai, Ujwala Patil, and Uma Mudenagudi (2020). "Learning-Based Hole Detection in 3D Point Cloud Towards Hole Filling." *Procedia Computer Science*.

Tancik, Matthew, Vincent Casser, Xinchen Yan, Sabeek Pradhan, Ben Mildenhall, Pratul P Srinivasan, Jonathan T Barron, and Henrik Kretzschmar (2022). "Block-NeRF: Scalable Large Scene Neural View Synthesis." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Tang, Pingbo, Daniel Huber, and Burcu Akinci (2007). "A Comparative Analysis of Depth-Discontinuity and Mixed-Pixel Detection Algorithms." *International Conference on 3-D Digital Imaging and Modeling (3DIM)*.

Tateno, Keisuke, Federico Tombari, and Nassir Navab (2016). "When 2.5D is not enough: Simultaneous Reconstruction, Segmentation and Recognition on dense SLAM." *IEEE International Conference on Robotics and Automation (ICRA)*.

Tchapmi, Lyne, Christopher Choy, Iro Armeni, JunYoung Gwak, and Silvio Savarese (2017). "SEGCloud: Semantic Segmentation of 3D Point Clouds." *International Conference on 3D Vision (3DV)*.

Tewari, Ayush, Justus Thies, Ben Mildenhall, Pratul Srinivasan, Edgar Tretschk, W Yifan, Christoph Lassner, Vincent Sitzmann, Ricardo Martin-Brualla, Stephen Lombardi, et al. (2022). "Advances in Neural Rendering." *Computer Graphics Forum (CGF)*.

Thies, Justus, Michael Zollhöfer, and Matthias Nießner (2019). "Deferred Neural Rendering: Image Synthesis using Neural Textures." *ACM Transactions on Graphics (TOG)*.

Thomas, Hugues, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas (2019). "KPConv: Flexible and Deformable Convolution for Point Clouds." *IEEE International Conference on Computer Vision (ICCV)*.

Tölle, Malte, Max-Heinrich Laves, and Alexander Schlaefer (2021). "A Mean-Field Variational Inference Approach to Deep Image Prior for Inverse Problems in Medical Imaging." *Medical Imaging with Deep Learning*.

Ulyanov, Dmitry, Andrea Vedaldi, and Victor Lempitsky (2018). "Deep Image Prior." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Vicini, Delio, Sébastien Speierer, and Wenzel Jakob (2021). "Path Replay Backpropagation: Differentiating Light Paths using Constant Memory and Linear Time." *ACM Transactions on Graphics (TOG)*.

Von Gioi, Rafael Grompone, Jeremie Jakubowicz, Jean-Michel Morel, and Gregory Randall (2008). "LSD: A Fast Line Segment Detector with a False Detection Control." *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*.

Wang, Lizhi, Chen Sun, Ying Fu, Min H Kim, and Hua Huang (2019a). "Hyperspectral Image Reconstruction Using a Deep Spatial-Spectral Prior." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Wang, Qian, Hoon Sohn, and Jack CP Cheng (2019b). "Development of high-accuracy edge line estimation algorithms using terrestrial laser scanning." *Automation in Construction*.

Wang, Xiaogang, Yuelang Xu, Kai Xu, Andrea Tagliasacchi, Bin Zhou, Ali Mahdavi-Amiri, and Hao Zhang (2020). "PIE-NET: Parametric Inference of Point Cloud Edges." *Advances in Neural Information Processing Systems (NeurIPS)*.

Ward, Gregory J (1992). "Measuring and Modeling Anisotropic Reflection." *Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*.

Weber, Christopher, Stefanie Hahmann, and Hans Hagen (2010). "Sharp Feature Detection in Point Clouds." *International Conference on Shape Modeling and Applications*.

Weber, Christopher, Stefanie Hahmann, Hans Hagen, and Georges-Pierre Bonneau (2012). "Sharp feature preserving MLS surface reconstruction based on local feature line approximations." *Graphical Models*.

Weinmann, Martin, Boris Jutzi, Stefan Hinz, and Clément Mallet (2015a). "Semantic point cloud interpretation based on optimal neighborhoods, relevant features and efficient classifiers." *ISPRS Journal of Photogrammetry and Remote Sensing (P&RS)*.

Weinmann, Martin, Alena Schmidt, Clément Mallet, Stefan Hinz, Franz Rottensteiner, and Boris Jutzi (2015b). "Contextual classification of point cloud data by exploiting individual 3D neigbourhoods." *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*.

Weinmann, Martin, Steffen Urban, Stefan Hinz, Boris Jutzi, and Clément Mallet (2015c). "Distinctive 2D and 3D features for automated large-scale scene analysis in urban areas." *Computers & Graphics*.

Whelan, Thomas, Michael Kaess, Maurice Fallon, Hordur Johannsson, John Leonard, and John McDonald (2012). "Kintinuous: Spatially Extended KinectFusion."

Whelan, Thomas, Michael Kaess, Hordur Johannsson, Maurice Fallon, John J Leonard, and John McDonald (2015). "Real-time large-scale dense RGB-D SLAM with volumetric fusion." *International Journal of Robotics Research*.

Whelan, Thomas, Renato F Salas-Moreno, Ben Glocker, Andrew J Davison, and Stefan Leutenegger (2016). "ElasticFusion: Real-time dense SLAM and light source estimation." *International Journal of Robotics Research*.

Williams, Francis, Teseo Schneider, Claudio Silva, Denis Zorin, Joan Bruna, and Daniele Panozzo (2019). "Deep Geometric Prior for Surface Reconstruction." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Wills, Josh, Sameer Agarwal, David Kriegman, and Serge Belongie (2009). "Toward a Perceptual Space for Gloss." *ACM Transactions on Graphics (TOG)*.

Wu, Hongzhi, Zhaotian Wang, and Kun Zhou (2015). "Simultaneous Localization and Appearance Estimation with a Consumer RGB-D Camera." *IEEE Transactions on Visualization and Computer Graphics (TVCG)*.

Wu, Hongzhi and Kun Zhou (2015). "AppFusion: Interactive Appearance Acquisition Using a Kinect Sensor." *Computer Graphics Forum (CGF)*.

X-RITE (2019). *Pantora material hub*. URL: `https://web.archive.org/web/20190424232441/https://www.xrite.com/categories/appearance/pantora-software`.

Xia, Shaobo and Ruisheng Wang (2017). "A fast edge extraction method for mobile LiDAR point clouds." *IEEE Geoscience and Remote Sensing Letters*.

Xiao, Runquan, Yanling Xu, Zhen Hou, Chao Chen, and Shanben Chen (2019). “An adaptive feature extraction algorithm for multiple typical seam tracking based on vision sensor in robotic arc welding.” *Sensors and Actuators A: Physical*. ISSN: 0924-4247. DOI: 10.1016/j.sna.2019.111533. URL: https://www.sciencedirect.com/science/article/pii/S0924424719307605.

Xie, Yuxing, Jiaojiao Tian, and Xiao Xiang Zhu (2020). “Linking points with labels in 3D: A review of point cloud semantic segmentation.” *IEEE Geoscience and Remote Sensing Magazine*.

Yagüe-Fabra, José Antonio, Sinué Ontiveros, Roberto Jiménez, Shahab Chitchian, Guido Tosello, and Simone Carmignato (2013). “A 3D edge detection technique for surface extraction in computed tomography for dimensional metrology applications.” *CIRP Annals*.

Yang, Xuhui, Yong Xu, Yuhui Quan, and Hui Ji (2020). “Image Denoising via Sequential Ensemble Learning.” *IEEE Transactions on Image Processing*.

Yu, Lequan, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng (2018). “EC-Net: an Edge-aware Point set Consolidation Network.” *European Conference on Computer Vision (ECCV)*.

Zhang, Cheng, Bailey Miller, Kan Yan, Ioannis Gkioulekas, and Shuang Zhao (2020). “Path-Space Differentiable Rendering.” *ACM transactions on graphics*.

Zhang, Fan, Shaodi You, Yu Li, and Ying Fu (2022). “HSI-Guided Intrinsic Image Decomposition for Outdoor Scenes.” *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zhang, Kai, Wangmeng Zuo, Shuhang Gu, and Lei Zhang (2017). “Learning Deep CNN Denoiser Prior for Image Restoration.” *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zhou, Qian-Yi and Vladlen Koltun (2014). “Color map optimization for 3d reconstruction with consumer depth cameras.” *ACM Transactions on Graphics (TOG)*.

Zoran, Daniel and Yair Weiss (2011). “From Learning Models of Natural Image Patches to Whole Image Restoration.” *IEEE International Conference on Computer Vision (ICCV)*.

List of Figures

List of Tables

Part IV

Appendix

Publication:
“Real-time Multi-material Reflectance Reconstruction for Large-scale Scenes under Uncontrolled Illumination from RGB-D Image Sequences”

Lukas Bode, Sebastian Merzbach, Patrick Stotko, Michael Weinmann, and Reinhard Klein

International Conference on 3D Vision (3DV)

2019

DOI: 10.1109/3DV.2019.00083

Real-time Multi-material Reflectance Reconstruction for Large-scale Scenes under Uncontrolled Illumination from RGB-D Image Sequences

Lukas Bode, Sebastian Merzbach, Patrick Stotko, Michael Weinmann, Reinhard Klein
Institute of Computer Science II
University of Bonn
{lbode,merzbach,stotko,mw,rk}@cs.uni-bonn.de

Abstract

Real-time reflectance reconstruction under uncontrolled illumination conditions is well-known to be a challenging task due to the complex interplay of scene geometry, surface reflectance and illumination. Nonetheless, recent works succeed in recovering both unknown reflectance and illumination in an uncontrolled setting. However, they are either limited regarding the scene complexity (single objects / homogeneous materials) or are not suitable for real-time applications. Our proposed method enables the recovery of heterogeneous surface reflectance (multiple objects and spatially varying materials) in complex scenes at real-time frame rates. We achieve this goal in the following way: First, we perform a 3D scene reconstruction from an input RGB-D stream in real-time. We then use a deep learning based method to estimate Ward BRDF parameters from observations gathered from individual segmented scene objects. Subsequently we refine these reflectance parameters to allow for spatial variations across the object surfaces. We evaluate our method on synthetic scenes and successfully apply it to real-world data.

1. Introduction

The digitization of scenes belongs to the classical computer vision tasks with numerous applications in entertainment, advertisement, cultural heritage as well as virtual and augmented reality. However, achieving realistic models relies on the accurate capture of the underlying properties such as geometry and reflectance characteristics which is complicated by the fact that only the interplay between surface geometry, material-specific reflectance characteristics and illumination conditions can be directly measured. Additional real-time constraints further complicate this task.

Regarding the separate real-time reconstruction of 3D scene geometry, impressive results have been reported with the aid of consumer RGB-D sensors such as the Kinect [31, 5, 43, 44, 12, 13, 6]. The decoupling of reflectance and illumination characteristics, however, remains a highly ill-posed challenge due to its severely under-constrained nature. As a result, many real-time reconstruction approaches rely on strong simplifications, such as using simple color textures to represent surface appearance. However, representing a surface point using a single color value is not sufficient. One needs to take into account that color observations incrementally captured for it may strongly vary due to view- and illumination dependent shadows or high-frequency illumination characteristics. Otherwise, such effects would be stored in the surface texture, which would lead to inconsistencies for scene relighting. To improve the quality of the reflectance reconstruction by separating the aforementioned effects in real-time, existing works exploit intrinsic image decomposition for (diffuse) albedo estimation [16, 11, 29, 26, 40]. These techniques achieve real-time capabilities at a reduced reconstruction accuracy. In contrast, estimating BRDF models together with the surrounding illumination with inverse rendering frameworks yields more accurate reconstructions that also take specular reflectance into account. Inverse rendering approaches utilize alternating optimizations of reflectance and illumination based on statistical priors [39, 22, 21, 3, 47, 23, 24, 37, 2]. However, the computational burden of these approaches prevents real-time performance. Other approaches have recently demonstrated impressive real-time reconstructions by leveraging markers and mirror spheres [48] or by using the potential of deep learning, even in the absence of HDR inputs [17, 28]. However, remaining limitations include the restriction of these BRDF estimation frameworks to single objects with homogeneous reflectance characteristics.

In this paper, we address these limitations by proposing a novel multi-material reflectance reconstruction framework for large-scale scenes with spatially varying surface characteristics under uncontrolled indoor illumination. This implies taking into account near-field illumination characteristics and extending previous frameworks [17, 28] to handle inhomogeneous reflectance characteristics as well

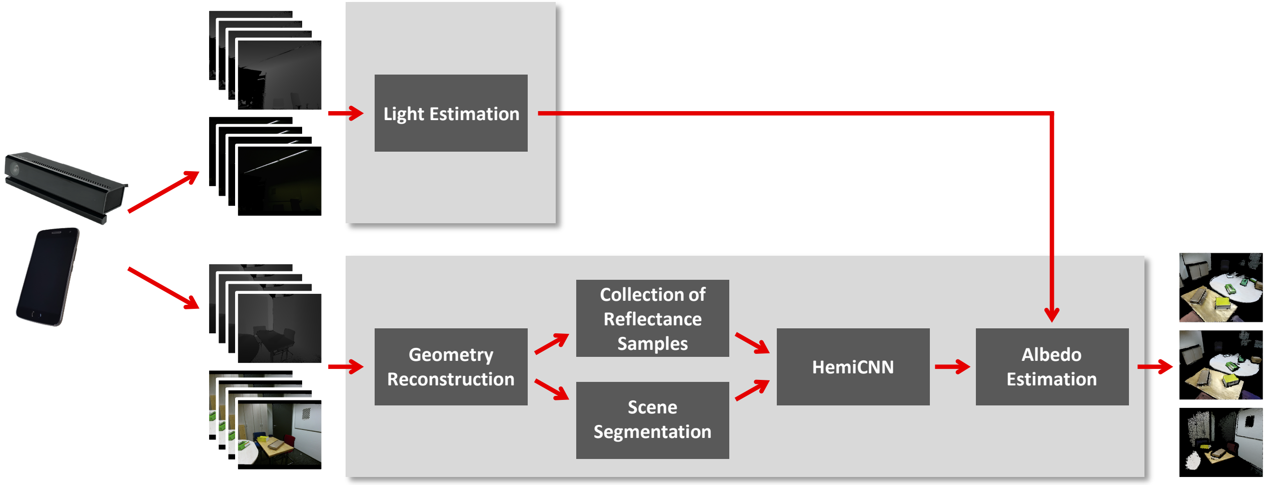

Figure 1. Overview of the proposed real-time multi-material acquisition approach.

as multiple materials in large scenarios in real-time. For this purpose, we capture near-field illumination characteristics, initially assuming that the illumination conditions in indoor scenarios remain constant during capture. In addition, the use of scene segmentation allows to associate the individual reflectance measurements to segments of homogeneous reflectance characteristics, so that within-segment observations can be exploited for the estimation of local surface reflectance behavior. In a final step, we estimate multi-material reflectance characteristics in terms of spatially varying parameters of the Ward BRDF based on the collected measurements utilizing the HemiCNN [17] with a subsequent refinement of diffuse albedo characteristics to allow handling spatially varying characteristics. Our evaluation demonstrates the potential of our approach in the scope of synthetic and real-world examples.

2. Related Work

Early work on separating reflectance and illumination includes in particular the intrinsic image decomposition [4], where an input image is decomposed into the product of a shading layer and a reflectance layer, and its numerous improvements since that time. However, the underlying representation based on two images is disadvantageous as the reflectance layer only represents the diffuse component while the specular component is stored together with the lighting in the shading layer.

Assuming known geometry, Haber et al. [10] and Diaz and Sturm [7] estimate Lambertian reflectance and illumination characteristics from images taken under uncontrolled conditions. Barron and Malik [3] estimate shape, reflectance, and illumination from a single image. Furthermore, using video frames as input, Dong et al. [8] exploit the knowledge regarding surface geometry of a rotating object to estimate spatially varying reflectance behavior and Palma et al. [33] captured SVBRFs while surrounding the object and approximating the environment with a few domination point light sources. In contrast, Wu and Zhou [48] applied the Kinect sensor as an active reflectometer in the IR spectrum and separately captured the illumination in the scene, which allows scanning the object geometry and appearance within several minutes while providing interactive visual feedback. Similarly, Knecht et al. [18] also explored the Kinect to estimate reflectance characteristics at interactive rates. In further work [47], color and depth images captured under unknown illumination serve as input to an offline joint optimization of camera poses, materials, illumination, and surface normals. On-the-fly reflectance estimation at interactive rates for objects exhibiting a homogeneous smooth surface reflectance behavior has been achieved by Kim et al. [17] based on a learned model trained on synthetic data. Solely considering flat material samples, Aittala et al. [1] exploit self-similarities in the surface reflectance behavior to fit spatially-varying BRDFs over a detailed normal map based on a flash/no-flash image pair depicting a flat material sample. Furthermore, Li et al. [19] infer BRDF characteristics for single images based on self-augmented convolutional neural networks.

Instead of assuming known surface geometry, several techniques [36, 9, 20, 25] use implicit shape priors and, hence, are tailored to objects used during the training.

Lombardi and Nishino [21, 24] employ priors for the reflectance model to extrapolate non-observed measurements in combination with illumination priors to jointly optimize for the reflectance and illumination characteristics. In subsequent work [23], this has been further improved to also handle complex scene appearance beyond single isolated objects. In all these approaches the considered objects are assumed to exhibit a smooth, homogeneous reflectance behavior and real-time performance has not been reached. Another offline estimation approach for illumination and material properties tailored to in the wild conditions has been proposed by Richter-Trummer et al. [38].

The recent work of Meka et al. [27] has been demonstrated to allow for live reflectance estimation from single images without assuming the aforementioned priors. This has been achieved based on the coupling of various encoder-decoder architectures to derive object segmentation, as well as more detailed reflectance information. However, the approach is tailored to the capture of single objects with homogeneous reflectance characteristics.

3. Multi-material Reflectance Estimation in Large Scenes from RGB-D Sequences

As illustrated in Figure 1, our framework for real-time multi-material reflectance reconstruction takes inputs in terms of RGB-D streams from commodity depth sensors such as the Microsoft Kinect or respective RGB-D sensors in smartphones. In an initial step, we recover the illumination characteristics in the scene (see Section 3.2). Thereby, we avoid the need for special calibration targets such as chrome spheres as used by Wu and Zhou [48]. Based on the initial illumination reconstruction, we then perform a real-time reflectance reconstruction by gathering view- and illumination-dependent observations for each surface point (Section 3.4), segmenting the scene into different objects (Section 3.5), and estimating material reflectance characteristics in terms of specular (Section 3.6) and diffuse albedo (Section 3.7). In Section 3.1, we first review the underlying reflectance representation and subsequently provide more details regarding the major components of our framework.

3.1. Image Formation and Reflectance Models

Before addressing the inverse rendering problem in terms of inferring surface reflectance characteristics, we briefly focus on the underlying image formation process that describes the light exchange at surfaces as described by the rendering equation [15]:

$$\begin{aligned} L_o(x,\omega_o) &= L_e(x,\omega_o) \\ &\quad + \int_{H_i} f_*(\omega_i,x,\omega_o) L_i(x,\omega_i)\cos\theta_i \, d\omega_i. \end{aligned} \quad (1)$$

The radiance L_o leaving some point x into direction ω_o is composed of the radiance L_e emitted from that point into direction ω_o, and the integral over the radiance L_i, incident at x from directions ω_i in the domain H_i, that gets reflected into direction ω_o according to a material-specific reflectance model f_*, weighted by the cosine of the angle between ω_i and the surface normal. Assuming that an object does not emit light on its own, we can ignore L_e.

In order to capture surface appearance, we have to recover the underlying reflectance, which is a severely ill-posed task difficult to solve in real-time. Therefore, following previous work, we assume that reflectance can be sufficiently described with parametric BRDF models [23, 47, 17]. Similar to Kim et al. [17], we use the Ward BRDF model [42]

$$f_{BRDF}(\omega_i,x,\omega_o) = \frac{\kappa_d(x)}{\pi} + \frac{\kappa_s(x)}{N}\cdot e^{\gamma}, \quad (2)$$

$$N = 4\pi\alpha^2\sqrt{\cos\theta_i\cdot\cos\theta_o}, \quad (3)$$

$$\gamma = -\frac{\tan{\theta_h}^2}{\alpha^2}, \quad (4)$$

as it can be seen as a trade-off between simplicity and the capability to represent a wide range of materials, and has been used in the domain of material perception [34, 46]. Here, κ_d denotes the diffuse and κ_s the specular albedo. The parameter α describes the surface roughness. Another common assumption is that each scene object consists of a single homogeneous material, such that it can be sufficiently described by the 7-dimensional Ward parameters. However, since very few real-world objects follow this assumption, we relax this assumption by performing a spatially varying albedo refinement. Finally, we ignore all indirect illumination effects like self-shadowing or interreflections.

3.2. Lighting Estimation

Knowledge of the illumination conditions in the scene facilitates the estimation of surface reflectance behavior and has been addressed e.g. by using special calibration targets, such as mirroring spheres, in front of the moving camera [48]. As we focus on indoor scenarios, we have to capture near-field illumination. Since time-of-flight sensors (e.g. the Microsoft Kinect v2) are not able to measure depth for mirror-like surfaces, we instead record illumination characteristics using a separate RGB-D image sequence capturing the light sources by direct observation. During this first recording, the sensor is configured to use a low exposure in order to achieve a clear separation of light sources from the remaining scene contents in the RGB images. Note that we do not need an additional RGB-D sensor as both image sequences can be recorded sequentially. We back project pixels of the RGB images with a luminance above a given threshold according to the corresponding depth data and apply a simple spatial mean-shift clustering for each frame individually. Fusing the resulting per image point light candidates over the whole sequence yields the final illumination configuration. Alternatively, voting-based approaches could be used [45, 33].

3.3. Geometry Reconstruction

Both the estimation of near-field illumination and reflectance rely on knowledge of the surrounding scene geometry. We use the VoxelHashing 3D reconstruction framework [32, 14] that allows real-time reconstruction of large scenes. It relies on an implicit voxel-based surface representation adapted to the underlying scene geometry. Instead of allocating voxels for the entire scene volume, a sparse set of voxel blocks managed by spatial hashing is used.

3.4. Local Collection of Reflectance Observations

The inference of surface reflectance characteristics relies on collecting local observations of surface appearance at each surface point under various viewing configurations per voxel and constant illumination conditions. Therefore, an observation is given as a pair of an RGB color value and a direction from which it has been observed. For every voxel in the hash table we determine the corresponding pixel in the depth image. By comparing the depth value with the distance between voxel and camera, we check whether the voxel is corresponding to some pixel in the RGB image or not. If the two values are sufficiently close, we sample the color from the RGB image and store it together with the voxel-to-camera direction as one observation. Observations that are too far from the surface or occluded are discarded.

Similar to the VoxelHashing framework, we store all those observations in a separate large observations pool in GPU memory and access them through a hash table which maps voxel coordinates to a list of observations. Holding the observations in GPU memory allows for efficient highly parallel acquisition and processing. The GPU memory, however, is already in high demand for the geometry reconstruction itself and the machine learning framework running the CNN (Section 3.6). Due to the large number of voxels in the scene and input image sequences that usually contain hundreds of frames, the memory consumption is a very limiting factor for this step. In order to keep the memory, as well as the computational requirements tractable, we introduce some optimizations:

First, we limit the number of stored observations for a single voxel to m, while ensuring that the most important reflectance characteristic are still captured. Therefore, we approximate a uniform sampling over the hemisphere in normal direction by discarding one of the two most similar observations when exceeding the limit after storing a new one. Experimentally we determine $m = 30$ to be a reasonable number of stored observations. This solution represents a trade-off between a low chance of missing valuable specularity information and computational complexity. Since this is a real-time pipeline, we set the focus on performance.

As a second optimization, we work on a coarser voxel grid for anything regarding the reflectance observations. Instead of the usual 8^3 voxels per voxel block used for the geometry reconstruction, we only use 2^3 or 4^3 voxels for a voxel block of the same spatial dimensions in this step. This downsampling is also the reason for using a separate voxel pool and hash table instead of directly integrating the observations in the geometry reconstruction voxel data structure. Separating the reflectance from the geometric observations additionally allows decoupling the geometry reconstruction from the material estimation framework.

3.5. Segmentation

Estimating multi-material reflectance is complicated by the fact that different materials may seem similar under certain viewing and illumination configurations. Instead of performing a color-based segmentation that may not distinguish material clusters correctly and connect distant dissimilar regions, we assume that the scene contains multiple objects with locally homogeneous materials. We therefore apply the depth-based segmentation by Tateno et al. [41]. It is based on the assumption that most objects have convex shapes, and thus tend to be separated by concave boundary regions in the depth maps. The concave regions are computed using the relative normal orientations from the depth maps and are segmented using connected component analysis. In addition, we exploit the temporal coherence of such regions over image sequences to make the segmentation consistent over time.

For further processing we need to be able to randomly sample voxels of a specific segment. In order to do this, we allocate a ring buffer of fixed size per material class, which is filled with voxel references utilizing the GPU.

3.6. Specular Material Parameter Estimation

For the material estimation, we assume every extracted segment to correspond to a region with homogeneous material characteristics. We thus have to predict one set of material parameters for the voxels assigned to a specific segment. For this purpose, we use the HemiCNN [17] to estimate specular albedo κ_s and the Ward roughness parameter α. While we use κ_s and α as provided by the HemiCNN, we use a novel albedo refinement technique to compute the diffuse albedo κ_d to increase robustness against violations of our homogeneity assumption, see Section 3.7.

In a first step of the estimation process, for every segment, we loop over its ring buffer containing the segment's voxels and randomly sample 25 of them. Per segment, we use those sampled voxels to create so called HemiImages from their reflectance observations. The observations' directions are rotated such that the z-axis is aligned with the surface normal, which is stored together with the reflectance observations. This results in the observations all being contained by the hemisphere in positive z direction. All directions are now projected onto the x-y-plane such that they are contained in the unit disk around the origin. To better preserve information under flat angles, we use a parabolic mapping instead of the orthogonal projection suggested by

Kim et al. [17]. The disk containing the projected observation directions is transformed to the range $[0;14]^2$. We subsequently use nearest neighbor interpolation on the observed colors to fill the pixel grid of 15×15 images. The created HemiImages are used as the input for HemiCNN.

We use a variation of the RMSE2 [17] as loss, i.e.

$$E(w,\hat{w}) = \lambda_d \left\| \begin{pmatrix} \lambda_l L - \lambda_l \hat{L} \\ a - \hat{a} \\ b - \hat{b} \end{pmatrix} \right\|_2^2 + \left\| \begin{pmatrix} c_r - \hat{c_r} \\ c_g - \hat{c_g} \\ c_b - \hat{c_b} \\ d - \hat{d} \end{pmatrix} \right\|_2^2 \tag{5}$$

with $w = (L, a, b, c_r, c_g, c_b, d)$ being the ground truth Ward parameters in a perceptually linear representation [35] and $\hat{w}$ analogously being the estimated parameters. For lower values for λ_d and λ_l the network focuses more on the specular estimation. Therefore, different than Kim et al. ($\lambda_d = \lambda_l = 1$), we use $\lambda_d = 0.1$ and $\lambda_l = 0.3$.

Since we estimate the scene's materials in real time, we have to run the material estimation step each frame. In order to reduce the susceptibility to noise in the individual material estimates, we fuse the material parameters over time.

Inspired by the truncated signed distance function (TSDF) update formula used in KinectFusion [30], we use an average over the materials for the single frames to temporally fuse local material parameter estimates, with a higher weight for current observations. However, instead of dividing by the number of material predictions after summing them up, we clamp the divisor (in our pipeline to 60).

3.7. Albedo Refinement

Applying the HemiCNN in a per-segment manner yields homogeneous diffuse and specular characteristics per segment. In order to relax this and address inhomogeneous reflectance characteristics, we refine the diffuse albedo while keeping the other modalities fixed, thereby allowing spatially varying surface appearance according to the observations in the voxel grid resolution.

Based on Equations 1 and 2, the k-th reflectance observation for one voxel can be expressed as

$$B_k = \sum_l \left(\frac{\kappa_d}{\pi} + \frac{\kappa_s}{N_l} \cdot e^{\gamma_l} \right) \cdot L_l \cdot \cos\theta_{i,l}, \tag{6}$$

where N_l, γ_l and $\theta_{i,l}$ are respectively the variables N, γ and θ_i for light source l. Solving for κ_d yields

$$\kappa_d = \pi \cdot \frac{B_k - \kappa_s \cdot \sum_l \frac{1}{N_l} \cdot e^{\gamma_l} \cdot L_l \cdot \cos\theta_{i,l}}{\sum_l L_l \cdot \cos\theta_{i,l}}. \tag{7}$$

For every frame we use the observations per voxel to recalculate the respective per-voxel diffuse albedo κ_d. Due to the approximately uniform sampling of the observations' directions, we achieve a high degree of temporal coherence by simply averaging the single estimates. The artifacts introduced by the rather low resolution of the observation voxel grid are reduced by applying trilinear interpolation.

4. Evaluation

After providing implementation details, we evaluate our technique for both synthetic and real-world scenarios.

4.1. Implementation Details

We performed all experiments using an Intel Core i7-4930K with 32 GB RAM and an Nvidia GeForce GTX 1080 with 8 GB VRAM. Following standard indoor 3D reconstruction approaches, we use a 3D space discretization with a resolution of 5 mm for the reconstructed model. Furthermore, we use grid resolutions of 2 cm, and 1 cm for the reflectance observations.

The data we use for training the HemiCNN is based on the SynBRDF [17] dataset. It contains 4432 RGB-D image sequences with 100 synthetic images per sequence, which show a single Ward-shaded object from different perspectives, illuminated using various environment maps. Due to its synthetic character, we know the ground truth Ward material parameters. The scenes are divided into 3574 training, 424 validation, and 434 test scenes. We use those images together with our previously described pipeline to create 500 HemiImages per sequence. Afterwards we sample 200 different random sets of 25 HemiImages to create 886400 labeled examples, on which we train our network. We train the HemiCNN in TensorFlow with the Adam optimizer, using a learning rate of 0.0001 and 150k batches, containing 32 examples each.

4.2. Synthetic Data

For synthetic data, we have direct access to the ground truth camera trajectory, segmentation, and material parameters. Our test scenarios consist of objects in a virtual scene and a camera moving around them in an oscillating manner. To generate such scenes, we utilize an OpenGL rasterization engine.

The benefits of our improved HemiCNN are shown in Figure 2. Using our modified HemiCNN allows to reconstruct the specular material characteristics more precisely (e.g. on the yellow bunny). Furthermore, our albedo refinement integrated into the material reconstruction pipeline also allows to reconstruct spatially varying diffuse albedos. A qualitative evaluation in Figure 3 shows that shading effects are mostly avoided in the refined diffuse albedo maps. Furthermore, the re-renderings with and without albedo refinement match the input RGB images closely for the cube sequence, where the individual objects are homogeneous. Figure 4 compares our reconstructed parameters to the ground truth on a synthetic scene.

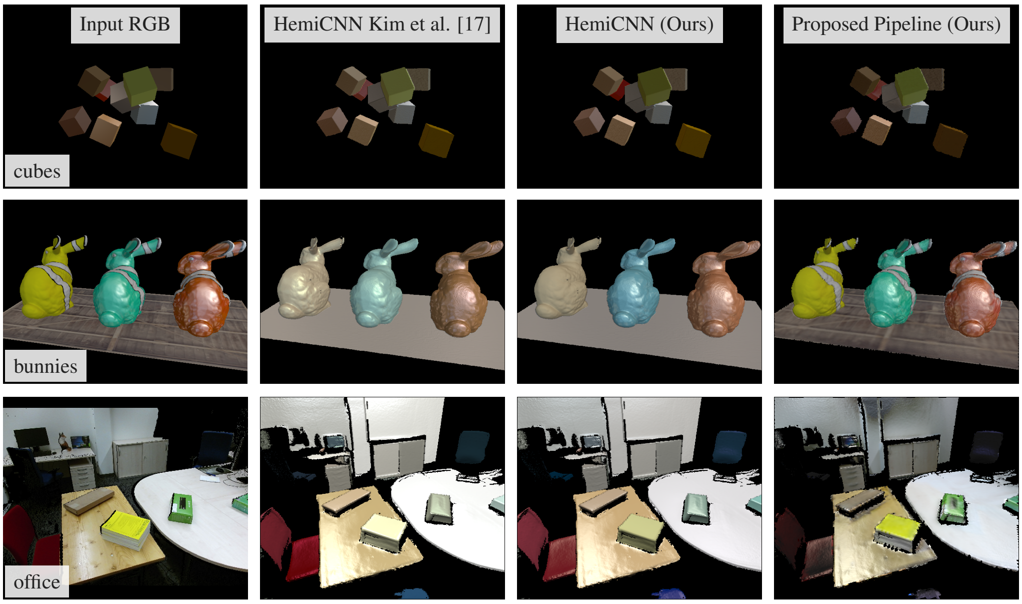

Figure 2. Comparison of our approach with the unmodified HemiCNN [17] on the cubes, bunnies, and office scenes. The first column shows the input RGB images, while the other columns show re-rendered RGB images reconstructed by the unmodified HemiCNN, HemiCNN with our proposed modifications, and our complete pipeline respectively. The reconstructions on the synthetic scenes use ground truth segmentation in order to focus the comparison on the material estimation aspect.

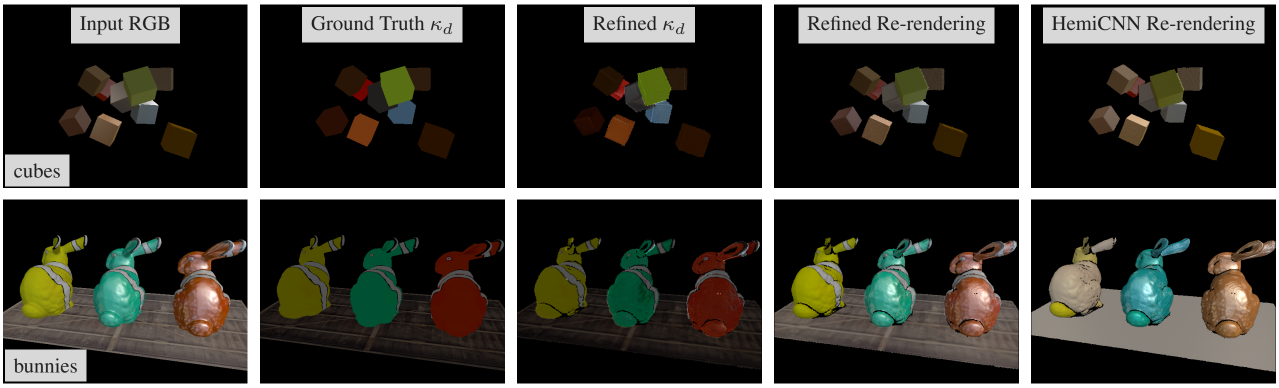

Figure 3. Results for two synthetic datasets: Image of the input sequence, ground truth diffuse albedo, refined diffuse albedo, scene re-rendering using all of the estimated Ward parameters and scene re-rendering using the diffuse albedo output of the HemiCNN directly (from left to right). In both cases, the distance between the scene's center and the camera is 4 m. The cubes have an edge length of 0.4 m and the bunnies have a height of 1.5 m.

The albedo refinement is particularly favorable for scenarios where the assumption of homogeneous materials is violated. Oscillations are induced by different viewpoints in the images. Additionally, the results in Figure 5 illustrate the influence of the albedo refinement for objects with spatially varying reflectance behavior. Further results regarding various error metrics are shown in Table 1.

4.3. Real-world Data

To demonstrate the performance of our technique on real-world data, we captured an indoor scene that contains a multitude of objects with different reflectance characteristics. For the RGB-D capturing of real-world scenes, we use the Microsoft Kinect v2 that delivers images with a resolution of 512×424 pixels at 30Hz.

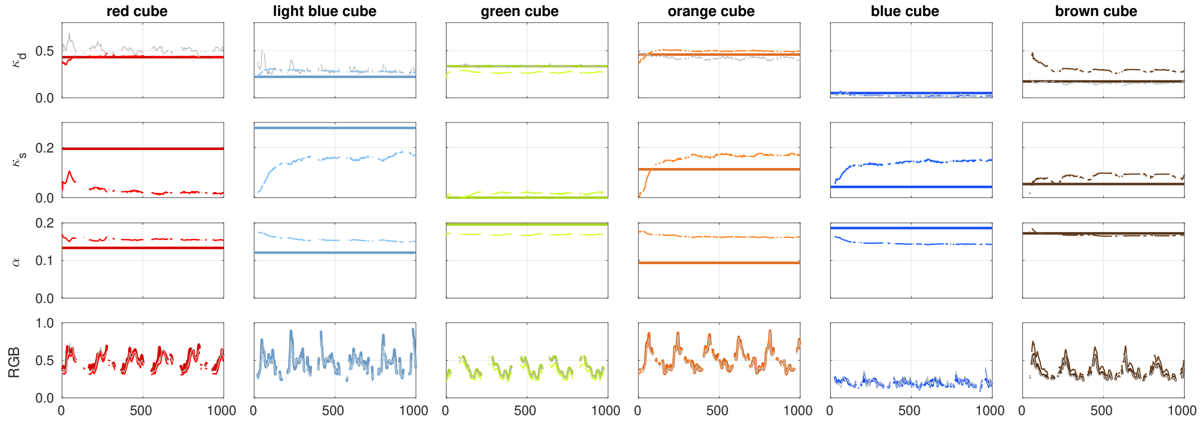

Figure 4. Quantitative evaluation over time (1000 frames) by comparison with ground truth BRDF parameters on a synthetic test scene over multiple objects. The first three rows show diffuse κ_d and specular albedo κ_s, as well as roughness parameter α. Plotted in bold are the ground truth BRDF parameters that are constant for each object (to avoid clutter, we plot only the red channel for κ_d and κ_s), the bottom row shows the red component of the rendered RGB color, where the bold line again represents ground truth. The thinner and lighter plots show our reconstructions. The gray plots show refined parameters. Gaps in the plots are caused by occlusions.

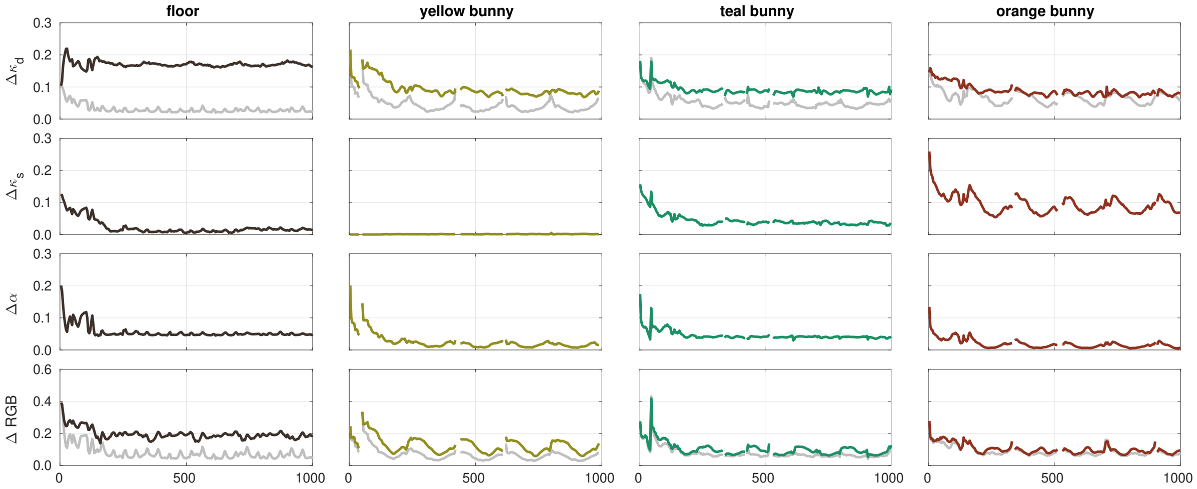

Figure 5. Quantitative evaluation over time (1000 frames) for the bunny dataset: L_1 errors (normalized over object region) of diffuse albedo (first row), specular albedo (second row), Ward roughness α, and final reconstruction error ΔRGB for the individual objects. The gray plots show the errors resulting from the refined model. Gaps in the plots are caused by occlusions.

	Δ_{L_1}					Δ_{L_2}					PSNR					SSIM				
	S1	S2	S3	S4	avg	S1	S2	S3	S4	avg	S1	S2	S3	S4	avg	S1	S2	S3	S4	avg
κ_d	0.17	0.09	0.09	0.09	0.11	0.18	0.13	0.12	0.11	0.13	13.36	19.23	20.80	19.94	18.33	0.89	0.95	0.96	0.93	0.93
$\kappa_{d,\text{ref}}$	**0.03**	**0.05**	**0.05**	**0.07**	**0.05**	**0.06**	**0.10**	**0.09**	**0.10**	**0.09**	**23.57**	**21.65**	**22.82**	**20.75**	**22.20**	**0.94**	**0.97**	**0.96**	**0.93**	**0.95**
κ_s	0.02	0.00	0.04	0.09	0.04	0.04	0.01	0.06	0.11	0.05	26.02	45.31	25.41	18.55	28.82	0.97	1.00	0.98	0.97	0.98
α	0.06	0.03	0.05	0.02	0.04	0.07	0.06	0.06	0.04	0.06	17.91	22.45	22.84	25.81	22.25	0.96	0.98	0.98	0.98	0.98
RGB	0.20	0.13	0.10	0.10	0.13	0.21	0.19	0.16	0.16	0.18	21.68	26.15	28.63	27.32	25.95	0.90	0.94	0.96	0.94	0.94
RGB$_{\text{ref}}$	**0.08**	**0.07**	**0.08**	**0.09**	**0.08**	**0.15**	**0.14**	**0.15**	**0.15**	**0.15**	**25.16**	**28.83**	**28.96**	**27.48**	**27.61**	**0.92**	**0.94**	**0.97**	**0.95**	**0.95**

Table 1. Different metric results averaged over the 1000 frames of the bunny dataset: mean absolute deviation (MAD, Δ_{L_1}), root mean square error (RMSE, Δ_{L_2}), peak signal to noise ratio (PSNR), structural similarity index (SSIM). The individual metrics are shown separately for the different scene objects (S1: floor, S2: yellow bunny, S3: teal bunny, S4: orange bunny), as well as averaged over the entire scene. The errors are computed on the individual model parameters (κ_d, κ_s and α), as well as the re-renderings (RGB). The second and the last row show the metrics based on the refined diffuse albedo ($\kappa_{d,\text{ref}}$) and corresponding re-renderings (RGB$_{\text{ref}}$). Highlighted in **bold** are the respective better results under each metric, showing that our refinements produce consistent improvements.

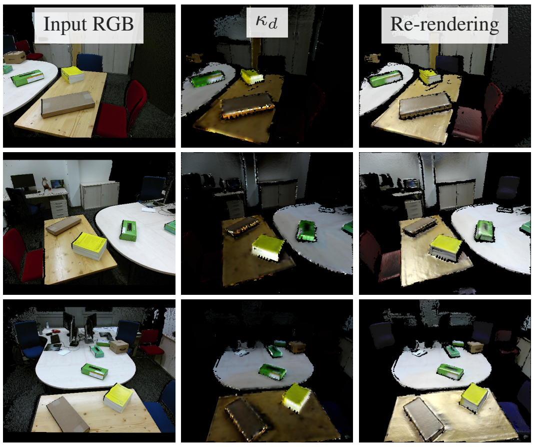

Figure 6. Results for a real-world office scene captured with the Microsoft Kinect v2 sensor: RGB input, estimated diffuse albedo, and Ward shaded re-rendering on three different frames.

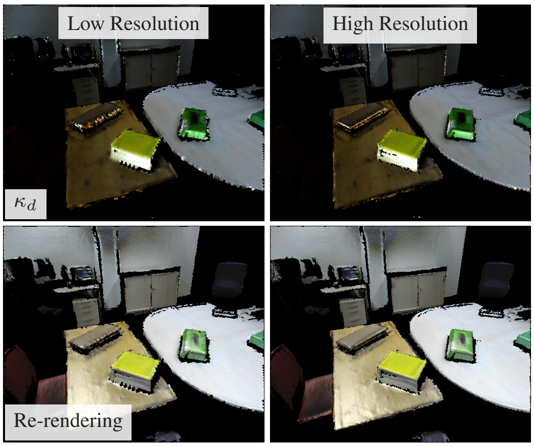

Figure 7. Comparison of results using different resolutions for the reflectance observations' voxel grid: 2 cm (left) and 1 cm (right).

As demonstrated in Figure 6, highlights are separated from the diffuse albedo, and the specular component is preserved in the reconstructed Ward parameters as illustrated by the re-rendering. Furthermore, Figure 7 shows that the reflectance observations' voxel grid resolution has only a minor effect on the re-renderings, albeit the difference being visible in the diffuse albedo maps.

4.4. Performance Evaluation

The timings needed by the individual components of our framework are shown in Table 2 and Table 3. As can be seen, our approach allows real-time material recovery on all low resolution test scenarios, as well as on the simple high resolution ones. Investigations about the α distribution in the data, as well as additional results on synthetic and real-world scenes are shown in the supplementary material.

Scene	cubes	bunnies	office
Geometry Rec.	3.480 ms	5.865 ms	8.906 ms
Refl. Obs. Collection	0.701 ms	1.474 ms	1.624 ms
Segmentation	1.154 ms	1.944 ms	1.755 ms
Specular Mat. Est.	5.997 ms	6.582 ms	6.039 ms
Albedo Refinement	5.962 ms	9.158 ms	11.438 ms
Total	17.294 ms	25.023 ms	29.763 ms

Table 2. Performance of the whole pipeline on various scenes with low (2 cm) voxel grid resolution for reflectance observations.

Scene	cubes	bunnies	office
Geometry Rec.	3.397 ms	5.867 ms	8.997 ms
Refl. Obs. Collection	2.965 ms	8.535 ms	9.967 ms
Segmentation	1.136 ms	1.980 ms	1.756 ms
Specular Mat. Est.	5.978 ms	6.507 ms	5.820 ms
Albedo Refinement	10.442 ms	21.579 ms	28.254 ms
Total	23.919 ms	44.467 ms	54.794 ms

Table 3. Performance of the whole pipeline on various scenes with high (1 cm) voxel grid resolution for reflectance observations.

4.5. Limitations and Future Work

Reconstructing scenes with high dynamic range (HDR) leads to problems with overexposure since consumer-grade RGB-D sensors like the Kinect typically only capture low dynamic range (LDR) images. This limitation could be tackled by augmenting the LDR inputs or reconstructing HDR from LDR images captured under varying exposures.

The sampling of the stored observations could be adapted to better match the object's specularity for sparsly observerd objects such as the chair and the wall in Figure 6. Further improvements include the optimization of our current implementation to allow refining the resolution of the reflectance observations and improving the segmentation by additionally considering albedo information.

5. Conclusion

In this paper, we presented a novel real-time multi-material reflectance reconstruction framework for large-scale scenes with spatially varying surface characteristics under uncontrolled static near-field indoor illumination. After an initial reconstruction of the near-field scene lighting, the framework uses the combination of real-time 3D reconstruction, scene segmentation and per-segment reflectance estimation. As demonstrated, our technique preserves specular characteristics in the estimated material parameters and additionally is capable of handling also spatially varying reflectance characteristics.

Acknowledgements

This work was supported by the DFG projects KL 1142/11-1 (DFG Research Unit FOR 2535 Anticipating Human Behavior) and KL 1142/9-2 (DFG Research Unit FOR 1505 Mapping on Demand).

References

[1] M. Aittala, T. Weyrich, and J. Lehtinen. Two-shot svbrdf capture for stationary materials. *ACM Trans. Graph.*, 34(4):110:1–110:13, July 2015.

[2] R. A. Albert, D. Y. Chan, D. B. Goldman, and J. F. O'Brien. Approximate svbrdf estimation from mobile phone video. In *Proceedings of the Eurographics Symposium on Rendering: Experimental Ideas & Implementations*, SR '18, pages 11–22, Goslar Germany, Germany, 2018. Eurographics Association.

[3] J. T. Barron and J. Malik. Shape, illumination, and reflectance from shading. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(8):1670–1687, Aug 2015.

[4] H. G. Barrow and J. M. Tenenbaum. *Recovering Intrinsic Scene Characteristics from Images*. Academic Press, 1978.

[5] J. Chen, D. Bautembach, and S. Izadi. Scalable Real-time Volumetric Surface Reconstruction. *ACM Trans. Graph.*, 32:113:1–113:16, 2013.

[6] A. Dai, M. Nießner, M. Zollhöfer, S. Izadi, and C. Theobalt. Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration. *ACM Trans. Graph.*, 36(4), May 2017.

[7] M. Díaz and P. Sturm. Estimating Photometric Properties from Image Collections. *Journal of Mathematical Imaging and Vision*, 47(1-2):93–107, Sept. 2013.

[8] Y. Dong, G. Chen, P. Peers, J. Zhang, and X. Tong. Appearance-from-motion: Recovering spatially varying surface reflectance under unknown lighting. *ACM Trans. Graph.*, 33(6):193:1–193:12, Nov. 2014.

[9] S. Georgoulis, K. Rematas, T. Ritschel, M. Fritz, L. J. V. Gool, and T. Tuytelaars. Delight-net: Decomposing reflectance maps into specular materials and natural illumination. *CoRR*, abs/1603.08240, 2016.

[10] T. Haber, C. Fuchs, P. Bekaert, H. Seidel, M. Goesele, and H. P. A. Lensch. Relighting objects from image collections. In *CVPR*, pages 627–634, 2009.

[11] M. Hachama, B. Ghanem, and P. Wonka. Intrinsic Scene Decomposition from RGB-D Images. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 810–818. IEEE Computer Society, 2015.

[12] O. Kähler, V. Prisacariu, J. Valentin, and D. Murray. Hierarchical Voxel Block Hashing for Efficient Integration of Depth Images. *IEEE Robotics and Automation Letters*, 1(1):192–197, 2016.

[13] O. Kähler, V. A. Prisacariu, and D. W. Murray. Real-Time Large-Scale Dense 3D Reconstruction with Loop Closure. In *European Conference on Computer Vision*, pages 500–516, 2016.

[14] O. Kähler, V. A. Prisacariu, C. Y. Ren, X. Sun, P. Torr, and D. Murray. Very High Frame Rate Volumetric Integration of Depth Images on Mobile Devices. *Visualization and Computer Graphics, IEEE Transactions on*, 21(11):1241–1250, 2015.

[15] J. T. Kajiya. The rendering equation. In *ACM Siggraph Computer Graphics*, volume 20, pages 143–150. ACM, 1986.

[16] C. Kerl, M. Souiai, J. Sturm, and D. Cremers. Towards Illumination-invariant 3D Reconstruction using ToF RGB-D Cameras. In *International Conference on 3D Vision (3DV)*, volume 1, pages 39–46. IEEE, 2014.

[17] K. Kim, J. Gu, S. Tyree, P. Molchanov, M. Nießner, and J. Kautz. A lightweight approach for on-the-fly reflectance estimation. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 20–28, 2017.

[18] M. Knecht, G. Tanzmeister, C. Traxler, and M. Wimmer. Interactive brdf estimation for mixed-reality applications. *Journal of WSCG*, 20(1):47–56, June 2012.

[19] X. Li, Y. Dong, P. Peers, and X. Tong. Modeling surface appearance from a single photograph using self-augmented convolutional neural networks. *ACM Trans. Graph.*, 36(4):45:1–45:11, July 2017.

[20] G. Liu, D. Ceylan, E. Yumer, J. Yang, and J.-M. Lien. Material editing using a physically based rendering network. *IEEE International Conference on Computer Vision (ICCV)*, pages 2280–2288, 2017.

[21] S. Lombardi and K. Nishino. Reflectance and natural illumination from a single image. In *Computer Vision - ECCV 2012 - 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part VI*, pages 582–595, 2012.

[22] S. Lombardi and K. Nishino. Single image multimaterial estimation. In *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, CVPR '12, pages 238–245, Washington, DC, USA, 2012. IEEE Computer Society.

[23] S. Lombardi and K. Nishino. Radiometric scene decomposition: Scene reflectance, illumination, and geometry from RGB-D images. In *Fourth International Conference on 3D Vision, 3DV 2016, Stanford, CA, USA, October 25-28, 2016*, pages 305–313, 2016.

[24] S. Lombardi and K. Nishino. Reflectance and illumination recovery in the wild. *IEEE Trans. Pattern Anal. Mach. Intell.*, 38(1):129–141, 2016.

[25] M. Maximov, T. Ritschel, and M. Fritz. Deep appearance maps. *CoRR*, abs/1804.00863, 2018.

[26] A. Meka, G. Fox, M. Zollhöfer, C. Richardt, and C. Theobalt. Live User-Guided Intrinsic Video For Static Scene. *IEEE Transactions on Visualization and Computer Graphics*, 23(11):2447–2454, 2017.

[27] A. Meka, M. Maximov, M. Zollhoefer, A. Chatterjee, H.-P. Seidel, C. Richardt, and C. Theobalt. Lime: Live intrinsic material estimation. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[28] A. Meka, M. Maximov, M. Zollhöfer, A. Chatterjee, H.-P. Seidel, C. Richardt, and C. Theobalt. LIME: Live Intrinsic Material Estimation. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, pages 6315–6324, June 2018.

[29] A. Meka, M. Zollhöfer, C. Richardt, and C. Theobalt. Live Intrinsic Video. *ACM Transactions on Graphics (TOG)*, 35(4):109:1–109:14, 2016.

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

[31] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger. Real-time 3D Reconstruction at Scale Using Voxel Hashing. *ACM Transactions on Graphics (TOG)*, 32(6):169:1–169:11, 2013.

[32] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger. Real-time 3d reconstruction at scale using voxel hashing. *ACM Transactions on Graphics (ToG)*, 32(6):169, 2013.

[33] G. Palma, M. Callieri, M. Dellepiane, and R. Scopigno. A statistical method for svbrdf approximation from video sequences in general lighting conditions. *Computer Graphics Forum*, 31(4):1491–1500, 2012.

[34] F. Pellacini, J. A. Ferwerda, and D. P. Greenberg. Toward a psychophysically-based light reflection model for image synthesis. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pages 55–64, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[35] F. Pellacini, J. A. Ferwerda, and D. P. Greenberg. Toward a psychophysically-based light reflection model for image synthesis. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 55–64. ACM Press/Addison-Wesley Publishing Co., 2000.

[36] K. Rematas, T. Ritschel, M. Fritz, E. Gavves, and T. Tuytelaars. Deep reflectance maps. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 4508–4516, 2016.

[37] T. Richter-Trummer, D. Kalkofen, J. Park, and D. Schmalstieg. Instant Mixed Reality Lighting from Casual Scanning. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 27–36. IEEE, 2016.

[38] T. Richter-Trummer, D. Kalkofen, J. Park, and D. Schmalstieg. Instant mixed reality lighting from casual scanning. In *2016 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 27–36, Sept 2016.

[39] F. Romeiro and T. Zickler. Blind reflectometry. In K. Daniilidis, P. Maragos, and N. Paragios, editors, *Computer Vision – ECCV 2010*, pages 45–58, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[40] P. Stotko, M. Weinmann, and R. Klein. Albedo estimation for real-time 3d reconstruction using rgb-d and ir data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 150:213–225, 2019.

[41] K. Tateno, F. Tombari, and N. Navab. When 2.5 d is not enough: Simultaneous reconstruction, segmentation and recognition on dense slam. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pages 2295–2302. IEEE, 2016.

[42] G. J. Ward. Measuring and modeling anisotropic reflection. In *Proceedings of the 19th Annual Conference on Computer Graphics and Interactive Techniques*, pages 265–272, 1992.

[43] T. Whelan, M. Kaess, M. Fallon, H. Johannsson, J. Leonard, and J. McDonald. Kintinuous: Spatially Extended Kinect-Fusion. In *RSS Workshop on RGB-D: Advanced Reasoning with Depth Cameras*, 2012.

[44] T. Whelan, M. Kaess, H. Johannsson, M. Fallon, J. J. Leonard, and J. McDonald. Real-time large-scale dense RGB-D SLAM with volumetric fusion. *The International Journal of Robotics Research*, 34(4-5):598–626, 2015.

[45] T. Whelan, R. F. Salas-Moreno, B. Glocker, A. J. Davison, and S. Leutenegger. ElasticFusion: Real-Time Dense SLAM and Light Source Estimation. *The International Journal of Robotics Research*, 35(14):1697–1716, 2016.

[46] J. Wills, S. Agarwal, D. Kriegman, and S. Belongie. Toward a perceptual space for gloss. *ACM Trans. Graph.*, 28(4):103:1–103:15, Sept. 2009.

[47] H. Wu, Z. Wang, and K. Zhou. Simultaneous Localization and Appearance Estimation with a Consumer RGB-D Camera. *IEEE Transactions on Visualization and Computer Graphics*, 22:2012–2023, 2016.

[48] H. Wu and K. Zhou. Appfusion: Interactive appearance acquisition using a kinect sensor. *Comput. Graph. Forum*, 34(6):289–298, Sept. 2015.

Publication:
"Locally-guided Neural Denoising"

Lukas Bode, Sebastian Merzbach, Julian Kaltheuner,
Michael Weinmann, and Reinhard Klein

Graphics and Visual Computing (GVC)

2022

DOI: 10.1016/j.gvc.2022.200058

Contents lists available at ScienceDirect

Graphics & Visual Computing

journal homepage: www.elsevier.com/locate/gvc

Locally-Guided Neural Denoising

Lukas Bode[a], Sebastian Merzbach[a], Julian Kaltheuner[a], Michael Weinmann[b], Reinhard Klein[a]

[a] University of Bonn, Friedrich-Hirzebruch-Allee 8, Bonn, 53115, Germany
[b] Delft University of Technology, Van Mourik Broekmanweg 6, Delft, 2628 XE, Netherlands

ARTICLE INFO

Article history:
Accepted 22 September 2022

Keywords: Computer Graphics, Image Processing, Denoising, SVBRDF Restoration

ABSTRACT

Noise-like artifacts are common in measured or fitted data across various domains, e.g. photography, geometric reconstructions in terms of point clouds or meshes, as well as reflectance measurements and the respective fitting of commonly used reflectance models to them. State-of-the-art denoising approaches focus on specific noise characteristics usually observed in photography. However, these approaches do not perform well if data is corrupted with location-dependent noise. A typical example is the acquisition of heterogeneous materials, which leads to different noise levels due to different behavior of the components either during acquisition or during reconstruction. We address this problem by first automatically determining location-dependent noise levels in the input data and demonstrate that state-of-the-art denoising algorithms can usually benefit from this guidance with only minor modifications to their loss function or employed regularization mechanisms. To generate this information for guidance, we analyze patchwise variances and subsequently derive per-pixel importance values. We demonstrate the benefits of such locally-guided denoising at the examples of the Deep Image Prior method and the Self2Self method.

1. Introduction

Data containing high levels of noise poses a huge problem for many applications in entertainment, advertisement, and design. Immersive experiences of scenes and objects rely on respective high-fidelity depictions and are significantly impacted by noisy data resulting from the capture or modeling process. Unfortunately, certain types and levels of noise cannot be avoided during data capture. Physical or economic constraints might affect the choice of the sensor or the amount and quality of the

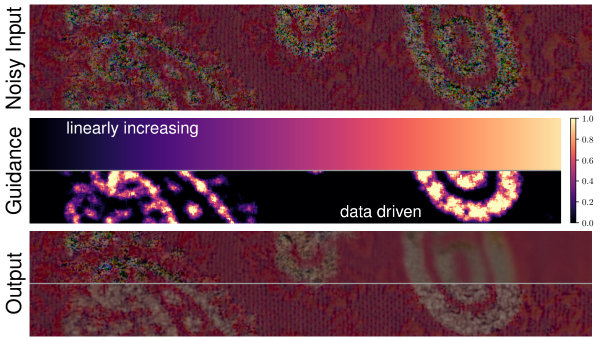

Fig. 1. We present a novel approach to remove spatially concentrated noise from images. Given a noisy input image (top row), a guidance map (middle row) can be used to control the denoising intensity of state-of-the-art denoising algorithms in a spatially-varying manner. We propose a fully automatic way to generate such guidance information by detecting noisy pixels in the input (middle row, bottom half). Hereby, corrupted pixels of the input can be denoised while preserving fine details in others (bottom).

e-mail: lbode@cs.uni-bonn.de (Lukas Bode), merzbach@cs.uni-bonn.de (Sebastian Merzbach), kaltheun@cs.uni-bonn.de (Julian Kaltheuner), M.Weinmann@tudelft.nl (Michael Weinmann), rk@cs.uni-bonn.de (Reinhard Klein)

data that can be handled while meeting requirements regarding the computational burden for a task. Therefore, methods are typically designed to be robust to noisy data. Whereas certain types of noise including sensor noise can typically be handled robustly, the robustness to other noise types including compression artifacts or missing data is often still lacking and relies on sophisticated denoising methods.

In the field of appearance capture and modeling – which is concerned with creating photo-realistic virtual models that capture details regarding surface geometry and reflectance behavior of their real-world counterparts – noisy 3D measurements, inaccurate calibration and image noise have to be dealt with. Oftentimes, these are corrupted by non-uniform location-dependent noise as depicted in Figure 1. The typical way to handle such data is to apply image restoration algorithms like super-resolution, denoising and inpainting, which aim at recovering an original image x from its corrupted version $\tilde{x}$. This can be stated in terms of the optimization problem

$$\arg\min_{x} E(x; \tilde{x}) + R(x) \tag{1}$$

with the data term $E(x; \tilde{x})$ and a regularization term $R(x)$. In contrast to the task-specific data term, finding a good prior $R(x)$ is challenging. For a surjective mapping $g : \theta \mapsto x$, Functional (1) corresponds to

$$\arg\min_{\theta} E(g(\theta); \tilde{x}) + R(g(\theta)). \tag{2}$$

As shown by Ulyanov et al. [1], the choice of a *good* (possibly injective) mapping g allows getting rid of the prior term. By defining $g(\theta)$ as $f_\theta(z)$, where f is given by a deep neural network with parameters θ and using a fixed input z, we obtain

$$\arg\min_{\theta} E(f_\theta(z); \tilde{x}), \tag{3}$$

which can be solved based on gradient descent, i.e. we optimize the neural network's parameters to finally represent the searched restored version of the image $x^* = f_{\theta^*}(z)$ based on the optimal network weights θ^*. In other words, the underlying inverse problem is regularized by the deep network itself. Other approaches [2, 3] combined this approach with additional priors.

These restoration methods like the Deep Image Prior are hard to control, which poses a problem in the above example of data corruped with location-dependent noise. Applying the Deep Image Prior approach without modification results in either loss of fine details or carrying over large amounts of artifacts.

In this paper, we propose a method to control the training of state-of-the-art learning-based denoising algorithms to enable successful restoration of such data. At the example of the restoration of fitted spatially-varying bidirectional reflectance distribution function (SVBRDF) textures that describe the reflectance behavior of surfaces, we show how characteristic properties of occuring artifacts can be leveraged to guide the optimization. In particular, we introduce spatially varying guidance by means of a per-pixel importance value, which can be calculated in a fully automatic manner by analyzing patch-wise variances in the image. Based on two exemplary denoising approaches [1, 4], we show how to utilize the per-pixel importance values during the denoising process with only minor modifications to the original algorithms (see Figure 1). We validate the potential of our approaches in comparison to the respective original algorithms without modifications as well as another learning-based state-of-the-art denoising method [5], where our approaches outperform the baselines in terms of image quality of the restored images.

2. Related Work

In the context of model-based optimization for inverse problems such as restoration, denoising, superresolution and deblurring, it is well-known that the typically involved regularization term has a significant influence on the resulting performance. Therefore, lots of effort has been spent on finding good denoiser priors. Total variation [6, 7] has been widely applied, but the results may exhibit watercolor-like artifacts. Further approaches include Gaussian mixture models (GMM) [8] and the computationally expensive K-SVD denoiser prior [9]. Furthermore, non-local means [10] as well as block-matching and 3D filtering (BM3D) [11] tend to oversmooth irregular structures for images that do not exhibit self-similarities. As leveraging the correlation between different color channels by jointly handling them has been shown to lead to better performance in comparison to the independent handling of color channels [12], several works focused on color priors (e.g. [13, 14, 15]). Popular techniques such as CBM3D [13] rely on first decorrelating the image into a luminance-chrominance color space and subsequently applying the gray BM3D method for each transformed color channel separately. However, the resulting luminance-chrominance color channels still remain correlated [16], which indicates that it might be beneficial to jointly handle RGB channels.

Instead of the aforementioned hand-designed approaches, recent work particularly focused on learning-based methods to find the respective color image priors capturing characteristics of the given data. The learned deep CNN denoiser prior by Zhang et al. [17] benefits from the parallelization of the inference on the GPU and exploits the prior modeling capacity offered by deep architectures. Building on this work, the denoising algorithm by Yang et al. [18] utilizes ensemble learning to improve on the results, while Quan et al. [19] designed a complex-valued CNN to leverage insights from classical image recovery algorithms. However, the approach involves a training on a large dataset of thousands of clean/noisy image pairs. Despite relying on an image dataset for training as well, Recorrupted-to-Recorrupted [20] lifted the requirement for clean images in the dataset by proposing to learn a mapping of corrupted images to other corrupted images following the same noise distribution but with the noise being independent of the noise in the input image. Consequently, the clean image can be found by the averaging of multiple corrupted images. In contrast, the untrained approach by Ulyanov et al. [1] on Deep Image Priors (DIPs) shows that low-level statistics of a single input image can be sufficiently captured by the structure of a single DIP generator network. Invariance to adversarial perturbations and the suppression of non-robust image features are particularly achieved in the early iterations [21] after which overfitting starts to occur. To avoid the need for early stopping, i.e. finding a suitable number of iterations where the image prior does not overfit to noise characteristics or artifacts, other works rely on Bayesian approaches [22, 23] or under-parametrization based on deep decoder approaches [24] to prevent the overfitting and reach a stable convergence behavior. Further work on DIPs focused on optimizing the underlying network architecture as part of the denoising process [25, 26]. The potential of such deep priors have also been demonstrated for hyperspectral image denoising [27, 28] and even for surface reconstruction [29, 30, 31].

A similar approach, which in contrast to DIP is not relying on early stopping, has been introduced with Self2Self by Quan et al. [4]. Instead of finding a mapping from fixed noise to the input image, they use a similar U-Net architecture to find a mapping from a noisy input image to a clean image directly. Regularization is handled by employing a Bernoulli input masking scheme as well as dropout in the decoder layers.

More recently, CVF-SID [5] has been proposed as an approach for self-supervised single image denoising by disentangling clean image, signal-dependent noise and signal-independent noise in an end-to-end fashion. In the field of nonblind image deconvolution, Chen et al. [32] introduced a spatially-adaptive dropout scheme to handle the solution ambiguity introduced by the deblurring problem. While also assuming the input image to be corrupted by Gaussian white noise, they rely on the assumption of the noise being uniformly distributed over the image and independent of the image signal in order to denoise the image during the deblurring process.

3. Methodology

The goal of our work is to widen the range of problems, commonly used learning-based self-supervised single image denoising algorithms can be successfully applied to. While not being the only use-case for our work, we are specifically targeting the problem of denoising images with an arbitrary number of channels corrupted with location-dependent noise instead of noise being uniformly distributed over the whole image. Current state-of-the-art algorithms tend to either introduce additional blurriness in originally clean pixels or are not capable of sufficiently removing the noise from the image.

We first propose the calculation of importance images based on an estimated per-pixel noise level. Subsequently, we present exemplary minor adjustments to the Deep Image Prior (DIP) as well as the Self2Self (S2S) method in order to guide their denoising process according to the importance values.

3.1. Inference of a Guidance Map

We propose the guidance of image processing operations like denoising based on a guidance map in terms of a per-pixel importance value $m(x, y)$ for pixel (x, y), where values close to 1 indicate that the pixel of the input image should be preserved in the denoised image while pixels with importance close to 0 should be denoised as much as possible as they are assumed to have a low signal-to-noise ratio. Note, that this importance is directly related to the noise level of a pixel via

$$m(x, y) = 1 - n(x, y), \tag{4}$$

where $n(x, y)$ is the noise level for pixel (x, y). While calculating the true noise level from the image signal is an underconstrained problem, for the purpose of the guidance map it suffices to find a rough estimate of it as we can rely on the natural regularization capabilities of the underlying denoising algorithms. If working with RGB images, noise level estimates are calculated independently for all channels. The maximum over the noise levels of all channels is calculated before the remapping step described in Section 3.1.3. For the remainder of this section, we assume to be working with greyscale images for notational simplicity.

3.1.1. Variance-based Noise Level Estimation

Building on the assumption that noisy regions usually have a high variance, the naive way would be to estimate the per-pixel noise level as patchwise variance of the respective pixel neighborhood. The variance for such a neighborhood $\mathcal{N}(x, y) \subseteq \mathcal{I}$ is defined as

$$n_{\text{var}}(x, y) = \frac{1}{|\mathcal{N}(x, y)|} \sum_{(x', y') \in \mathcal{N}(x, y)} (\mathcal{I}(x', y') - \mu(\mathcal{N}(x, y)))^2 \tag{5}$$

and the mean over an arbitrary set of pixels $\mathcal{P}$ is defined as

$$\mu(\mathcal{P}) = \frac{1}{|\mathcal{P}|} \sum_{(x', y') \in \mathcal{P}} \mathcal{I}(x', y'). \tag{6}$$

However, this noise level estimate is prone to erroneously high values at discontinuities in the input image which we typically want to preserve in the denoised image making this method applicable only for very smooth images.

3.1.2. SVD-based Noise Level Estimation

To alleviate the aforementioned problem and allow for better adaptation to local noise characteristics, we apply a local noise level estimation. For this purpose, we propose to split the pixel neighborhood $\mathcal{N}(x, y)$ into two disjoint subsets $\mathcal{N}_{\text{lower}}(x, y)$ and $\mathcal{N}_{\text{upper}}(x, y)$ depending on whether the respective pixel is below or above the patch mean $\mu(\mathcal{N}(x, y))$, such that

$$\mathcal{N}(x, y) = \mathcal{N}_{\text{lower}}(x, y) \cup \mathcal{N}_{\text{upper}}(x, y) \tag{7}$$

and

$$\mathcal{N}_{\text{lower}}(x, y) \cap \mathcal{N}_{\text{upper}}(x, y) = \emptyset. \tag{8}$$

Subsequently, pixels of both subsets are lifted into $\mathbb{R}^3$

$$\mathcal{N}^3_{\{\text{lower,upper}\}}(x, y) = \left\{ \left. \begin{pmatrix} x' \\ y' \\ \mathcal{I}(x', y') \end{pmatrix} \right| (x', y') \in \mathcal{N}_{\{\text{lower,upper}\}}(x, y) \right\}, \tag{9}$$

where $\cdot_{\{a,b\}}$ combines equations for $\cdot_a$ and $\cdot_b$ for notational simplicity. Afterwards, matrices $M_{\mathcal{N}^3_{\{\text{lower,upper}\}}}$ can be constructed to perform an singular value decomposition (SVD) (dependence on the pixel (x, y) omitted for notational simplicity)

$$M_{\mathcal{N}^3_{\{\text{lower,upper}\}}} = U_{\{\text{lower,upper}\}} \Sigma_{\{\text{lower,upper}\}} V^T_{\{\text{lower,upper}\}}, \tag{10}$$

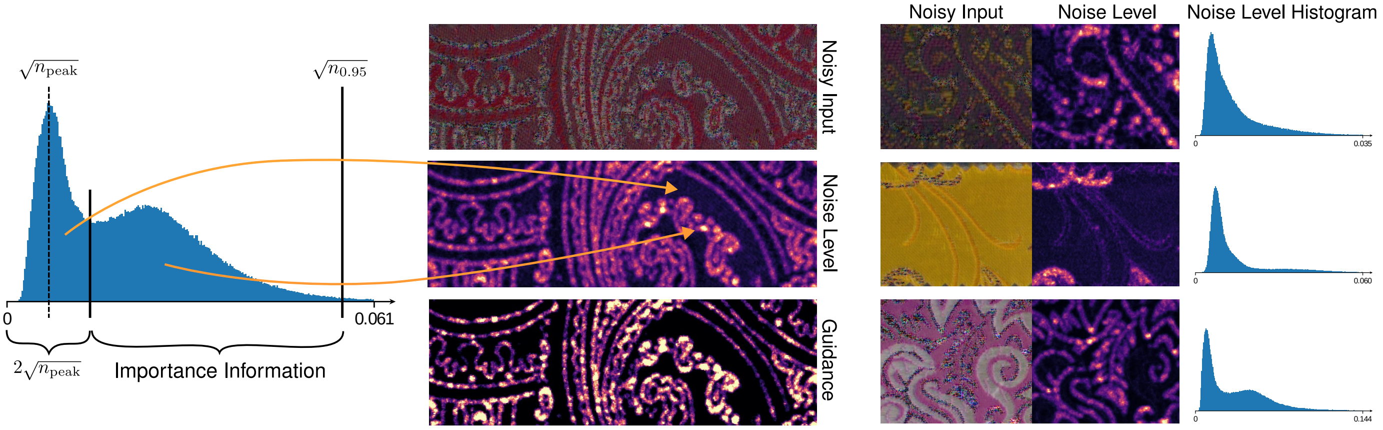

Fig. 2. Remapping procedure: Square root of estimated noise levels, i.e. the standard deviation of pixel values, roughly follows a Gaussian distribution for noise-free pixels. By finding the peak of the distribution, the respective pixels can be discarded (left) and the remaining part up to the 95th percentile is remapped to $[0, 1]$, resulting in a robust guidance map (middle). This holds for other examples as well (right).

where $\sigma_{\{\text{lower,upper}\},i} = \Sigma_{\{\text{lower,upper}\},ii}$, i.e. the diagnoal entries of matrices $\Sigma_{\{\text{lower,upper}\}}$, are the singular values. We are looking for the smallest singular value

$$\sigma_{\{\text{lower,upper}\},\min} = \min_{i\in\{1,2,3\}} \sigma_{\{\text{lower,upper}\},i} \tag{11}$$

of each subset as this value can be interpreted as the variance, and therefore the amount of noise, of the subset in normal direction of a plane fitted to the respective pixels. As this analysis is conducted for both subsets of pixels individually, the approach is robust against image discontinuities in constrast to relying on the patchwise variance directly. Additionally, due to the SVD, smooth color gradients are not detected as noise either. These two partial noise level estimates can be reduced to an estimate for the whole patch by choosing an appropriate reduction operator. Experiments have shown that the results are best using the minimum of σ_{lower} and σ_{upper}. We argue, that an additional robustness against detecting high-frequency details in the image as noise is more important than additional accuracy in estimating the noise level. The noise level can therefore be estimated as

$$n_{\text{svd}}(x, y) = \min_{s\in\{\text{lower,upper}\}} \sigma_{s,\min}(x, y). \tag{12}$$

3.1.3. Remapping

Despite noisy pixels having usually higher estimated noise level values $n_{\{\text{var,svd}\}}(x, y)$, we can still observe significant values for clean image pixels as well. Non-zero noise level estimates might prevent the full overfitting of the denoising network to clean pixels and thus can introduce unwanted blurriness for respective pixels. To avoid this, we apply a remapping technique to generate the final guidance images.

We observed that the square root of the estimated noise levels, i.e. the standard deviation of pixel values, for clean pixels roughly follows a Gaussian distribution as depicted in Figure 2. By calculating a histogram over the noise levels of all pixels, we find the bin with the highest pixel count as this is assumed to be the peak of the distribution with mean value $\sqrt{n_{\text{peak}}}$. Remapping our estimated noise level values using $(2\sqrt{n_{\text{peak}}})^2 = 4n_{\text{peak}}$ as lower bound and the 95th percentile $n_{0.95}$ as upper bound and clamping to $[0, 1]$ finally yields robust guidance images

$$m_{\{\text{var,svd}\}}(x, y) = 1 - \frac{n_{\{\text{var,svd}\}} - 4n_{\text{peak}}}{n_{0.95} - 4n_{\text{peak}}}. \tag{13}$$

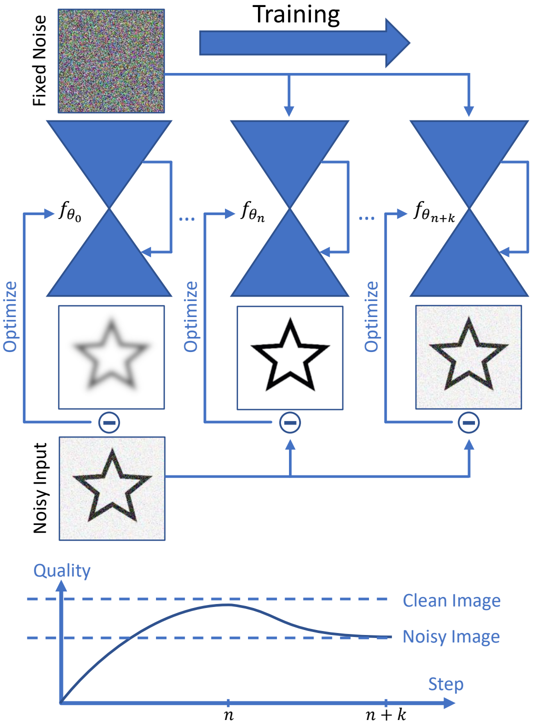

Fig. 3. Original DIP: Based on a noisy input image, a modified U-Net [33] is trained to map a fixed noise image to the noisy input image itself. Over the course of the training, natural image content is learned first due to the inherent regularization capabilities of the network. Early stopping is applied to stop the training process as soon as maximum quality is reached. If the network is trained further, the network output will finally converge to the actual noisy input image.

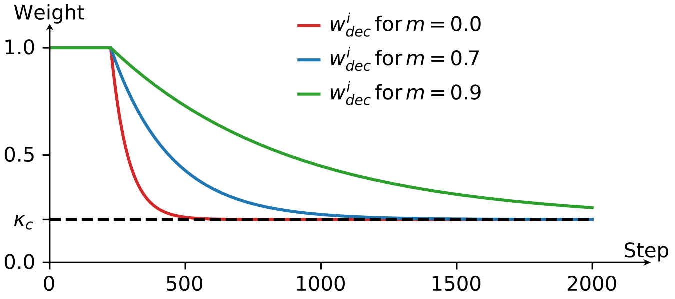

Fig. 4. Decaying per-pixel weight for DIP enables to locally control the denoising effect. After an initial warm-up phase (here $\kappa_w = 225$) with full denoising intensity, the weight decays exponentially (here $\kappa_r = 0.15$) and converges against a fixed lower bound (here $\kappa_c = 0.2$ for visualization purposes but set to $\kappa_c = 0.02$ during all experiments).

3.2. *Guided Deep Image Prior*

The generated guidance images can easily be used in state-of-the-art denoising algorithms, such as DIP [1]. This approach uses a neural network as natural prior for image restoration tasks including denoising. As depicted in Figure 3, the input for the network is a fixed noise image with 32 channels consisting of uniformly distributed random numbers in the range $[0, 0.1]$. Iteratively, the network learns a mapping from the noise image to the noisy input image by minimizing an $\mathcal{L}_2$-loss. During this training process, the network learns the more natural low-frequency components of the noisy input image first, while the high-frequency components are only learned in later stages. Hence, by interrupting the training process before the network learns to reconstruct the unwanted noise, we can consider the network output as denoised image. For further regularization, random noise drawn from the normal distribution $N(0, 1/30)$ is added to the network input in each step to further regularize the training process.

As in the original approach, the output of the network is smoothed over multiple iterations with an exponential weight according to

$$\mathcal{I}^i = 0.01\hat{\mathcal{I}}^i + 0.99\mathcal{I}^{i-1}, \qquad (14)$$

where $\mathcal{I}^i$ is the output image in iteration i and $\hat{\mathcal{I}}^i$ is the actual network prediction. This way, artifacts accidentally produced by the trained network are mostly smoothed out resulting in more accurate restorations.

Where not stated differently, we are using the same hyperparameters as the original approach in the denoising setting. In particular, we thus configure the network to have an encoder and a decoder each consisting of five double convolution layers with 128 filters. Each double convolution also contains batch normalizations and LeakyReLU activation functions. Reflection padding is used as it is described to work best by the authors [1].

Using the standard $\mathcal{L}_2$-loss as proposed by Ulyanov et al. [1] results in missing fine details in clean parts while the artifacts are already being learned by the network in corrupted ones and therefore being carried over to the output image. Since the artifacts are potentially restricted to some parts of the noisy input image due to systematic reasons, we propose a guided loss function to have further control over the restoration process.

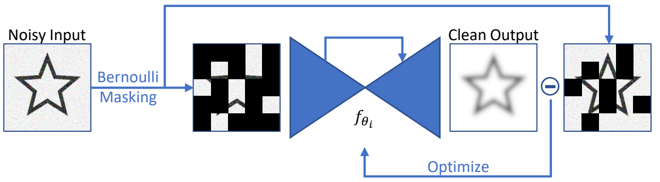

Fig. 5. Original S2S: A modified U-Net [33] is trained as an autoencoder to map a noisy input image to itself. During the training, two different regularization mechanisms are applied: First, bernoulli masking is performed to split the noisy image into a subset of pixels used as network input and the other pixels being used as target image in the loss function. This way, the loss is calculated only on pixels unseen by the network in the respective iteration. Second, dropout in the decoder layers of the network further help to prevent the autoencoder to learn the noise.

We use guidance image described in Section 3.1 for this purpose. Depending on a pixel's importance we stop the training process early by weighting down the loss induced by the respective pixel according to a weight $w^i_{\text{dec}}(x, y)$. This weight depends on the current iteration number i reducing the respective pixels contribution to the loss over time. The resulting loss term is

$$\mathcal{L}^i_{\text{dec}} = \frac{1}{|\mathcal{I}|}\sum_{(x,y)\in\mathcal{I}}((\hat{\mathcal{I}}^i(x,y) - \mathcal{I}(x,y))\cdot w^i_{\text{dec}}(x,y))^2. \qquad (15)$$

The decay weight $w^i_{\text{dec}}(x, y)$ is chosen to have an exponential fall-off after an initial warm-up phase with full contribution to the loss (dependence on the pixel (x, y) omitted for notational simplicity):

$$w^i_{\text{dec}} = \begin{cases} 1 & i < \kappa_w \\ (0.9 + 0.1m)^{(i-\kappa_w)\cdot\kappa_r}\cdot(1-\kappa_c) + \kappa_c & \text{else} \end{cases}, \qquad (16)$$

where κ_w specifies the number of initial warm-up iterations without any decay of the weight, while κ_r controls the decay rate and κ_c specifies a lower bound for the contribution of a single pixel. Note that we designed w^i_{dec} to converge to κ_c instead of 0 in order to avoid artifacts where the guidance image does not fit the degenerate areas perfectly. We rely on the natural image prior property of the network itself to prevent overfitting to noise for these pixels. We are using $\kappa_w = 225$, $\kappa_r = 0.15$ and $\kappa_c = 0.02$ for all our experiments. Corresponding plots are depicted in Figure 4.

3.3. *Guided Self2Self*

The second exemplary algorithm we are taking a closer look at here is S2S [4]. Similarly to DIP, this algorithm relies on the inherent regularization capabilities offered by neural networks. In contrast to DIP though, as depicted in Figure 5, S2S uses an U-Net like network to map the noisy image to the restored image. To prevent overfitting, the authors add two additional regularization mechanisms: masking of the network input image and loss as well as dropout in the decoder layers.

The masking is performed by applying Bernoulli sampling to the input image such that we get a mask containing a 1 with probability p_m and a 0 otherwise. Thus, this mask can be used to divide the pixels of the input image in two subsets. Before the image is given to the network, it is multiplied by the mask.

Therefore, only the first subset of pixels is contributing to the network output. Additionally, the L2 loss is modified to only use pixels which were not visible to the network in the respective iteration. Intuitively, the network is optimized to predict unseen pixels as close to the noisy input image as possible.

The dropout in decoder layers is another means of regularization. In contrast to most approaches relying on dropout, it is not deactivated in test mode. Instead, the network is evaluated multiple times with random dropout to simulate the training of multiple separate networks and averaging of the respective results. The authors have shown that this improves the quality of the resulting images.

Similarly to our modified DIP approach we are using the original authors' architecture and hyperparameters where not explicitly stated otherwise.

Analogously to DIP, S2S results in unwanted bluriness in clean regions which is a problem for our setting of spatially concentrated noise. Additionally, we have no control over the denoising intensity, which can be problematic if the signal-to-noise ratio is low in the input image. To alleviate these issues, we propose a generalization of S2S to allow the utilization of our guidance information as well as an additional denoising intensity parameter.

As in the original algorithm, the goal is to generate two binary image masks - $\mathcal{M}_i$ for masking the network input image and $\mathcal{M}_t$ for masking the difference image in the loss function. The underlying key idea here is to make the network overfit pixels with high-importance but denoise the image where importance is low. We achieve this by sampling the binary masks based on per-pixel probabilities $p_{\{i,t\}}$ defined as

$$p_i = p_{\text{imp}} \cdot 1 + (1 - p_{\text{imp}}) \cdot p_m \cdot p_d \tag{17}$$

$$p_t = p_{\text{imp}} \cdot 1 + (1 - p_{\text{imp}}) \cdot (1 - p_m) \cdot p_d. \tag{18}$$

with

$$p_{\text{imp}}(x, y) = m(x, y)^{\kappa_m}, \tag{19}$$

controlling the overfitting to high-importance image parts based on the guidance image m. Furthermore, κ_m controls the denoising strength for pixels with medium importance values (we set $\kappa_m = 2$ in all experiments) and p_d is a probability to discard a pixel completely from both masks to further increase the denoising effect. Where not stated explicitly, we use $p_d = 0.01$ for our experiments.

Intuitively, resulting masks can be thought of as linear interpolation between no masking happening at all, i.e. $\mathcal{M}_{\{i,t\}} = \mathcal{I}$, and standard S2S input masking according to p_{imp} with an additional probability p_d of pixels being considered neither in network input nor in the loss calculation.

The original S2S approach samples disjoint input and target masks. To replicate this behavior in our generalization, for each pixel j we have to handle four separate cases during sampling with their respective probabilities:

$$\Pr(j \in \mathcal{M}_i \wedge j \in \mathcal{M}_t) = p_b = p_{\text{imp}} \tag{20}$$

$$\Pr(j \in \mathcal{M}_i \wedge j \notin \mathcal{M}_t) = p_i = p_m \cdot p_d \cdot (1 - p_{\text{imp}}) \tag{21}$$

$$\Pr(j \notin \mathcal{M}_i \wedge j \in \mathcal{M}_t) = p_t = (1 - p_m) \cdot p_d \cdot (1 - p_{\text{imp}}) \tag{22}$$

$$\Pr(j \notin \mathcal{M}_i \wedge j \notin \mathcal{M}_t) = p_n = 1 - (p_b + p_i + p_t). \tag{23}$$

Note that setting $p_{\text{imp}} = 0$ and $p_d = 1$ yields the original S2S algorithm.

To ensure that the network is able to overfit high importance pixels, we also modify the dropout used in the decoder layers to use a modified dropout weight

$$\hat{p}_{\text{dropout}} = p_{\text{imp}} \cdot 0 + (1 - p_{\text{imp}}) \cdot p_{\text{dropout}} \tag{24}$$

per neuron. For inner decoder layers with lower resolution, downsampled importance images are used accordingly.

4. Results

4.1. Test Data

We evaluate the potential of our guided denoising approach using the diffuse textures of 14 different SVBRDFs produced by the fitting network of Merzbach et al. [34]. The measurements for all of these materials are publically available in the UBOFAB19 database [34].

The UBOFAB19 database uses the Geisler-Moroder variant [35] of the Ward BRDF [36] with Schlick's Fresnel approximation term as this model is expressive enough for a large variety of real-world materials. The model is parameterized based on the shading normal $\mathbf{n}_s \in \mathbb{R}^3$, the diffuse albedo $\mathbf{a}_d \in \mathbb{R}^3$, the specular albedo $\mathbf{a}_s \in \mathbb{R}^3$, the lobe roughness parameters $\sigma_x \in \mathbb{R}$ and $\sigma_y \in \mathbb{R}$, the anisotropy angle $\alpha \in \mathbb{R}$ and the 0-inclination reflection coefficient $F_0 \in \mathbb{R}$. We only apply our restoration process to the diffuse textures, since these are responsible for most of the artifacts in the final renderings. The diffuse textures of Pantora [37] SVBRDF fits as depicted in the first row of Figure 7 are considered to be the ground truth as they are also used as labels for training the fitting network of Merzbach et al. Note, however, that the SVBRDFs fitted by the network do not only contain a high amount of noise but are also very likely to be biased. Therefore, we cannot expect to achieve perfect results indistinguishable from ground truth using only image restoration methods.

4.2. Noise Level Estimation

Resulting estimated noise levels of two different textures using the naive patchwise variance and the more sophisticated SVD-based approach are shown in Figure 6. Both methods successfully assign high noise levels to actually noisy regions in the image. While the naive approach already performs well, the SVD-based algorithm reduces unwanted high values at discontinuities significantly. This is clearly visible in the upper regions of the yellow image, where the abrupt transition of yellow to grey pixels yields high patchwise variance values but low SVD-based noise levels as the individual subsets $\mathcal{N}_{\text{upper}}$ and $\mathcal{N}_{\text{lower}}$ can be approximated well by a plane. As the SVD-based noise level estimation performs better than the naive method without meaningful disadvantages, we are using the former for all denoising experiments.

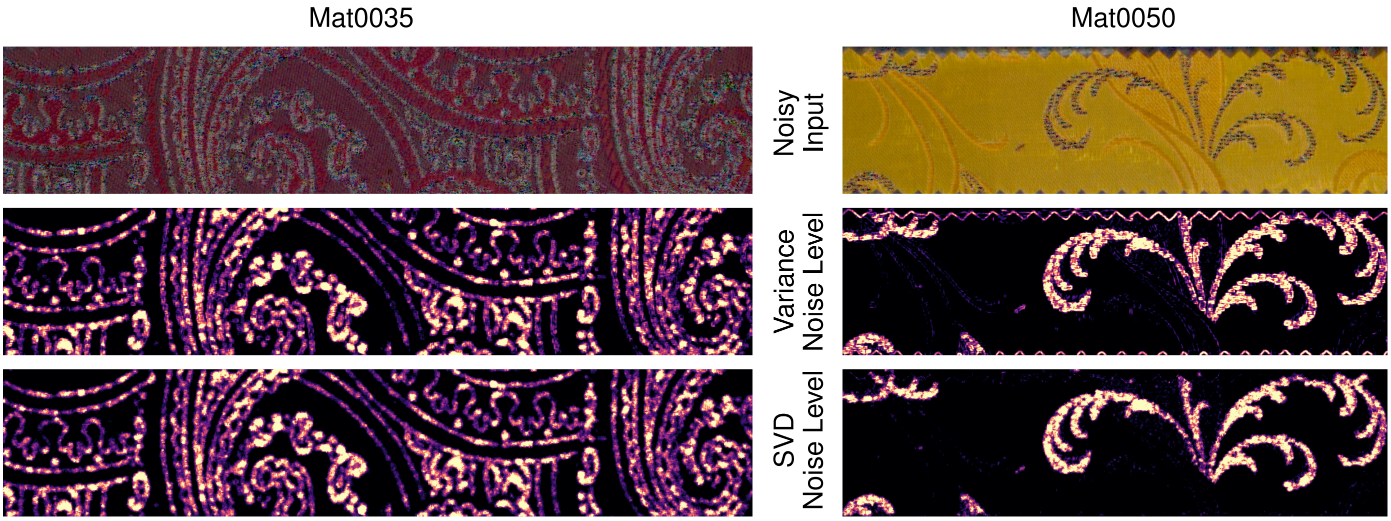

Fig. 6. Estimated noise level for two different images (top row) containing spatially concentrated noise. Using only the patchwise variance (middle row) erroneously yields high values for discontinuities in the image. Conducting the SVD-based analysis of the patches (bottom row) helps to filter out these errors.

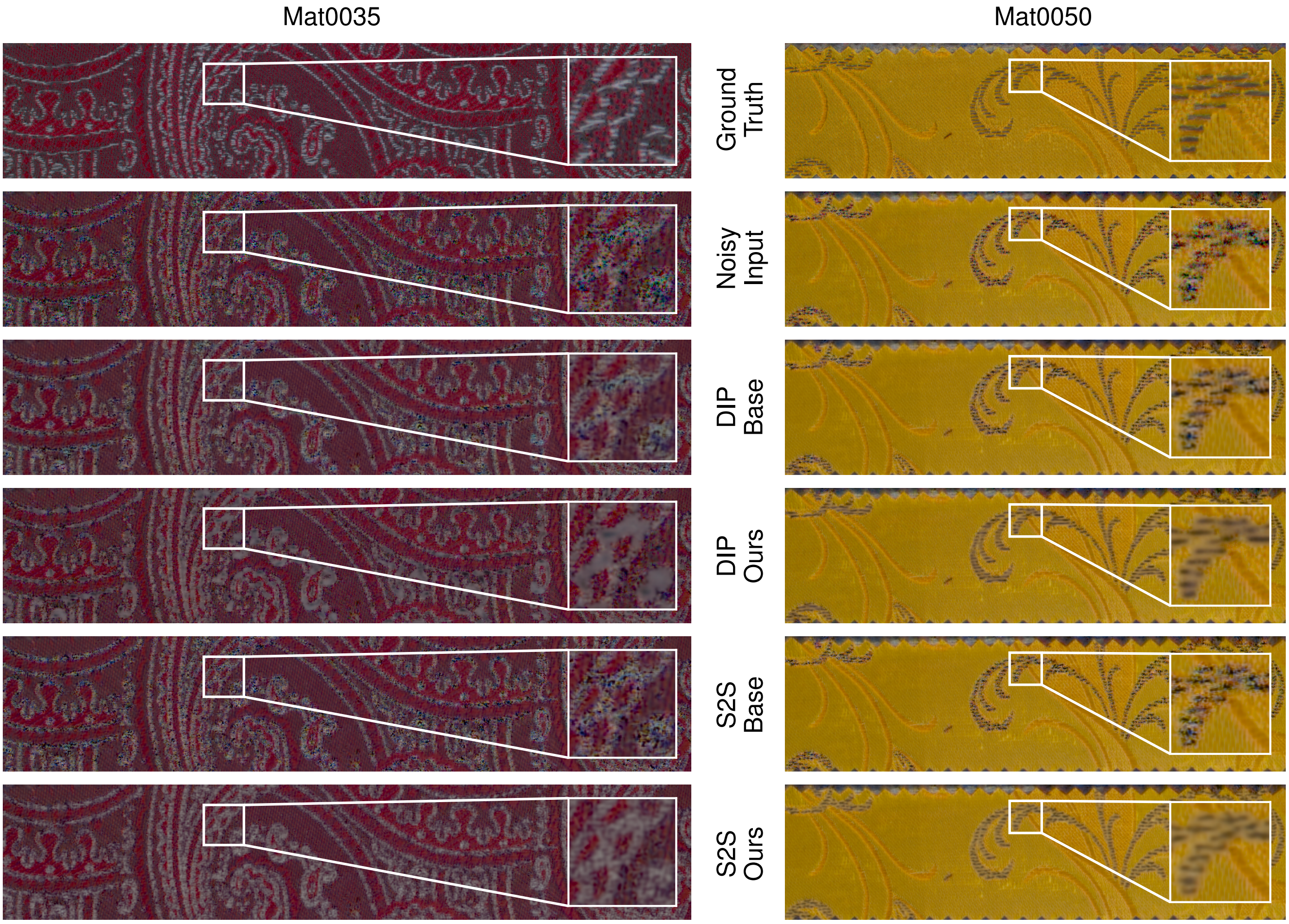

Fig. 7. Comparison of the results of the original DIP and S2S approaches with our modified variants. First row: Diffuse texture of the Pantora SVBRDF fit. Second row: Network-fitted textures used as input for all tested algorithms. Third and fifth row: Original denoising algorithms DIP and S2S. Fourth and sixth row: Out modified versions of the DIP and S2S algorithms using SVD-based guidance images.

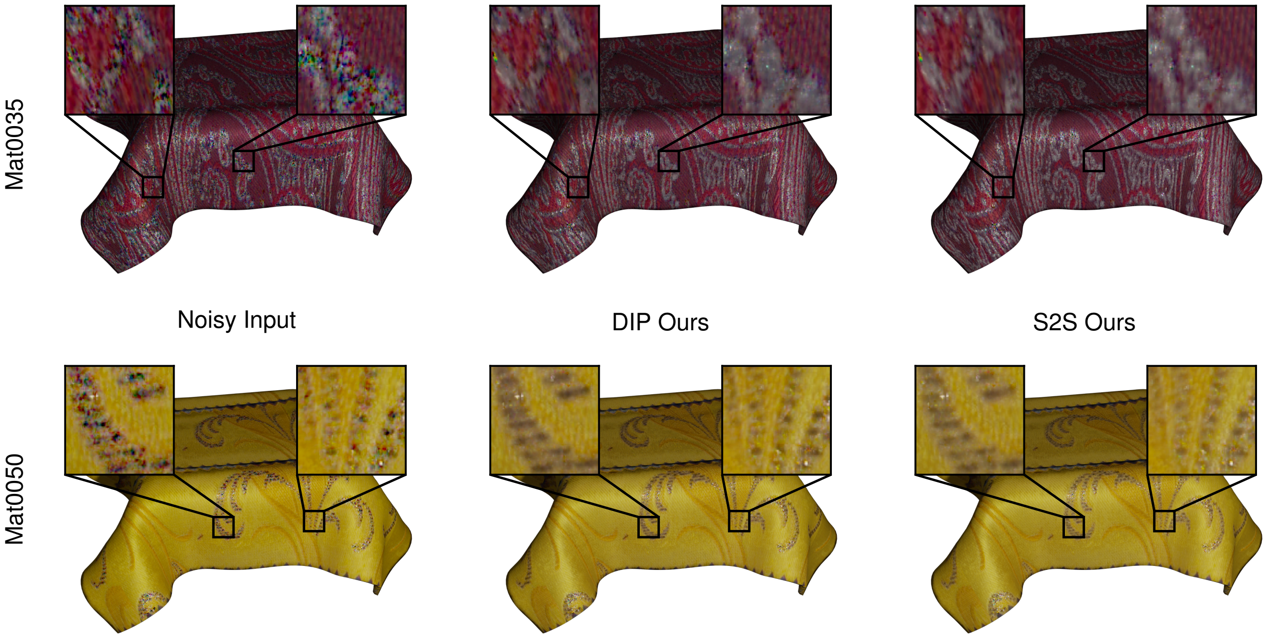

Fig. 8. Renderings of two different SVBRDFs fitted by the fitting network of Merzbach et al. [34] without denoising (left), denoised diffuse texture using our guided DIP variant (middle) and denoised diffuse texture using our guided S2S variant (right).

Algorithm	PSNR(↑)	SSIM(↑)
Input	25.9294	0.5437
CVF-SID [5]	27.8025	0.6085
DIP-Base [1]	26.8042	0.5846
DIP-Ours	27.5713	**0.6265**
S2S-Base [4]	27.4815	0.6149
S2S-Ours	**27.8747**	0.6197

Table 1. Quantitative comparison of several denoising approaches on a set of 14 diffuse textures of network fitted SVBRDFs using the diffuse textures of the respective Pantora fits as ground truth.

4.3. Denoising

Figure 7 depicts denoising results on two different textures. The network fitted textures contain strong noise artifacts especially in shiny areas of the captured fabric. Both, DIP and S2S, fail to remove these artifacts in a satisfactory manner, while at the same time blurring out the clean structure of the fabric. The modified algorithms are able to produce images which are mostly clean of colorful artifacts, but for the red fabric, our DIP variant seems to remove more details than necessary. More fine details are preserved using the guided S2S approach. On the yellow fabric, both of our algorithms produce similar results clearly outperforming their original counterpart respectively. The strong denoising effect of our guided approaches can also be seen in the rendered SVBRDFs in Figure 8. The results rendered with denoised diffuse albedo textures are containing much less disturbing colorful artifacts.

A quantitative comparison can be found in Table 1 comparing our guided denoising methods with CVF-SID [5], DIP [1] and S2S [4] on a dataset of 14 different images. We are calculating PSNR and mean SSIM for all images and average the respective values. For both images, higher values are better. Our guided denoising algorithms not only outperform their original complement, but also perform slightly better than another state-of-the-art learning-based denoising algorithm.

Additional results on natural images and a completely different texture are depicted in Figure 9. We used $p_d = 1$ for these experiments. In contrast to the original S2S method, our guided approach is able to preserve fine details in clean pixels.

4.4. Limitations

While being able to distinguish between clean and noisy image parts well in our examples, it is easy to construct artificial scenarios in which our SVD-based noise level estimation fails. However, our results suggest, that it should work well in practice as we can fallback to the natural regularization capabilities of the denoising methods.

We are able to adaptively denoise a partially noisy image using our guided denoising algorithms and therefore are able to overcome some of the original approach's limitations, but other problems with the respective approaches remain untouched. As the original DIP, our guided version is still relying on hyperparameter tuning. Despite working out for our examples, the optimal number of training iterations and the choice of w^i_{dec} control parameters might not be the same for every noisy input image, which might also be the reason for oversmoothing of noisy regions for the red fabric in Figure 7. Similarly, independent of being guided or not, S2S has to be tuned to the variance of the expected noise since it was not able to remove strong noise out of the box.

Finally, Figure 9 suggests, that the guided algorithms might produce slightly stronger artifacts in noisy pixels in comparison to the original approaches for some images. However, depending on the use-case, this is preferable over loss of details in

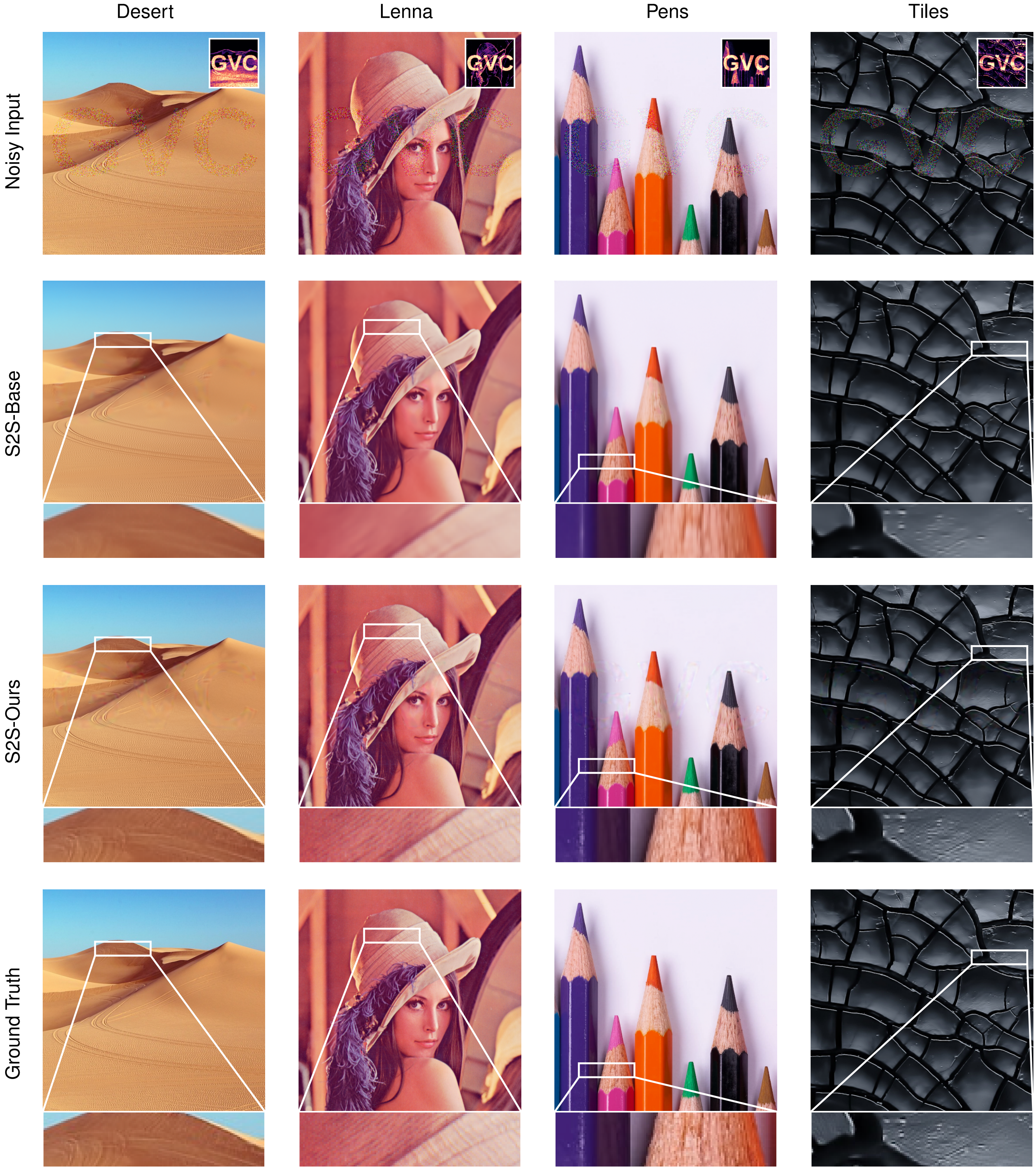

Fig. 9. Comparison of S2S-Base (second row) and S2S-Ours (third row) on three natural images and one texture corrupted with spatially concentrated white noise (first row, inset shows guidance map used for S2S-Ours). We used $p_d = 1$ for these experiments.

clean pixels.

5. Conclusion and Future Work

In this work, we have shown the limitations of off-the-shelf denoising algorithms regarding their capability of handling images which contain location-dependent noise-like artifacts. We proposed a novel method for detecting such noisy pixels and utilizing this additional information to guide state-of-the-art learning-based denoising approaches with only minor modifications. Depending on the nature of the underlying denoising approach, the generated guidance images can be used to either stop the training process early for parts of the image while continuing the training process in others, or it can be used to guide stochastic regularization approaches. By incorporating this additional guidance information, the resulting denoising algorithms were able to beat their original counterparts as well as outperform another state-of-the-art denoising algorithm.

Since our results suggest that other denoising algorithms could benefit from our guidance information in a similar manner, this should be tested as part of future research.

References

[1] Ulyanov, D, Vedaldi, A, Lempitsky, V. Deep image prior. In: CVPR. 2018, p. 9446–9454.

[2] Mataev, G, Milanfar, P, Elad, M. DeepRED: Deep image prior powered by RED. In: ICCV Workshops. 2019,.

[3] Liu, J, Sun, Y, Xu, X, Kamilov, US. Image restoration using total variation regularized deep image prior. In: International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE; 2019, p. 7715–7719.

[4] Quan, Y, Chen, M, Pang, T, Ji, H. Self2Self with dropout: Learning self-supervised denoising from single image. In: CVPR. 2020, p. 1890–1898.

[5] Neshatavar, R, Yavartanoo, M, Son, S, Lee, KM. CVF-SID: Cyclic multi-variate function for self-supervised image denoising by disentangling noise from image. In: CVPR. 2022, p. 17583–17591.

[6] Chambolle, A. An algorithm for total variation minimization and applications. Journal of Mathematical Imaging and Vision 2004;20(1):89–97.

[7] Osher, S, Burger, M, Goldfarb, D, Xu, J, Yin, W. An iterative regularization method for total variation-based image restoration. Multiscale Modeling & Simulation 2005;4(2):460–489.

[8] Zoran, D, Weiss, Y. From learning models of natural image patches to whole image restoration. In: ICCV. IEEE; 2011, p. 479–486.

[9] Elad, M, Aharon, M. Image denoising via sparse and redundant representations over learned dictionaries. IEEE Transactions on Image Processing 2006;15(12):3736–3745.

[10] Buades, A, Coll, B, Morel, JM. A non-local algorithm for image denoising. In: CVPR; vol. 2. IEEE; 2005, p. 60–65.

[11] Dabov, K, Foi, A, Katkovnik, V, Egiazarian, K. Image denoising by sparse 3-D transform-domain collaborative filtering. IEEE Transactions on Image Processing 2007;16(8):2080–2095.

[12] Foi, A, Katkovnik, V, Egiazarian, K. Pointwise shape-adaptive DCT denoising with structure preservation in luminance-chrominance space. In: Int. Workshop on Video Processing and Quality Metrics for Consumer Electronics. 2006,.

[13] Dabov, K, Foi, A, Katkovnik, V, Egiazarian, K. Color image denoising via sparse 3D collaborative filtering with grouping constraint in luminance-chrominance space. In: Int. Conference on Image Processing; vol. 1. IEEE; 2007,.

[14] Mairal, J, Elad, M, Sapiro, G. Sparse representation for color image restoration. IEEE Transactions on Image Processing 2008;17(1):53–69.

[15] Rajwade, A, Rangarajan, A, Banerjee, A. Image denoising using the higher order singular value decomposition. TPAMI 2012;35(4):849–862.

[16] Miyata, T. Inter-channel relation based vectorial total variation for color image recovery. In: International Conference on Image Processing (ICIP). IEEE; 2015, p. 2251–2255.

[17] Zhang, K, Zuo, W, Gu, S, Zhang, L. Learning deep CNN denoiser prior for image restoration. In: CVPR. 2017, p. 3929–3938.

[18] Yang, X, Xu, Y, Quan, Y, Ji, H. Image denoising via sequential ensemble learning. IEEE Transactions on Image Processing 2020;29:5038–5049.

[19] Quan, Y, Chen, Y, Shao, Y, Teng, H, Xu, Y, Ji, H. Image denoising using complex-valued deep CNN. Pattern Recognition 2021;111:107639.

[20] Pang, T, Zheng, H, Quan, Y, Ji, H. Recorrupted-to-Recorrupted: Unsupervised deep learning for image denoising. In: CVPR. 2021, p. 2043–2052.

[21] Kattamis, A, Adel, T, Weller, A. Exploring properties of the deep image prior. In: NeurIPS 2019 Posters. 2019,.

[22] Cheng, Z, Gadelha, M, Maji, S, Sheldon, D. A bayesian perspective on the deep image prior. In: CVPR. 2019, p. 5443–5451.

[23] Tölle, M, Laves, MH, Schlaefer, A. A mean-field variational inference approach to deep image prior for inverse problems in medical imaging. In: Proceedings of Machine Learning Research. 2021,.

[24] Heckel, R, Hand, P. Deep decoder: Concise image representations from untrained non-convolutional networks. In: ICLR. 2019,.

[25] Chen, YC, Gao, C, Robb, E, Huang, JB. Nas-dip: Learning deep image prior with neural architecture search. arXiv preprint arXiv:200811713 2020;.

[26] Ho, K, Gilbert, A, Jin, H, Collomosse, J. Neural architecture search for deep image prior. arXiv preprint arXiv:200104776 2020;.

[27] Sidorov, O, Yngve Hardeberg, J. Deep hyperspectral prior: Single-image denoising, inpainting, super-resolution. In: ICCV Workshops. 2019, p. 0–0.

[28] Wang, L, Sun, C, Fu, Y, Kim, MH, Huang, H. Hyperspectral image reconstruction using a deep spatial-spectral prior. In: CVPR. 2019, p. 8032–8041.

[29] Gadelha, M, Wang, R, Maji, S. Shape reconstruction using differentiable projections and deep priors. In: ICCV. 2019, p. 22–30.

[30] Williams, F, Schneider, T, Silva, C, Zorin, D, Bruna, J, Panozzo, D. Deep geometric prior for surface reconstruction. In: CVPR. 2019, p. 10130–10139.

[31] Hanocka, R, Metzer, G, Giryes, R, Cohen-Or, D. Point2Mesh: A self-prior for deformable meshes. arXiv preprint arXiv:200511084 2020;.

[32] Chen, M, Quan, Y, Pang, T, Ji, H. Nonblind image deconvolution via leveraging model uncertainty in an untrained deep neural network. International Journal of Computer Vision 2022;:1–20.

[33] Ronneberger, O, Fischer, P, Brox, T. U-net: Convolutional networks for biomedical image segmentation. In: Int. Conference on Medical Image Computing and Computer-assisted Intervention. Springer; 2015, p. 234–241.

[34] Merzbach, S, Hermann, M, Rump, M, Klein, R. Learned fitting of spatially varying BRDFs. In: Comp. Graph. Forum. Wiley Online Library; 2019, p. 193–205.

[35] Geisler-Moroder, D, Dür, A. A new ward BRDF model with bounded albedo. In: Comp. Graph. Forum; vol. 29. Wiley Online Library; 2010, p. 1391–1398.

[36] Ward, GJ. Measuring and modeling anisotropic reflection. In: Proceedings of the 19th annual conference on Computer graphics and interactive techniques. 1992, p. 265–272.

[37] X-RITE, . Pantora material hub. 2019. URL: `https://web.archive.org/web/20190424232441/https://www.xrite.com/categories/appearance/pantora-software`.