Scalable 3D Reconstruction for Immersive Virtual Reality Applications

Dissertation

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

vorgelegt von

Patrick Stotko

aus
Euskirchen

Bonn 2022

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

Gutachter:	Prof. Dr. Reinhard Klein
Gutachter:	Prof. Dr. Matthias B. Hullin

Tag der Promotion:	14.12.2022
Erscheinungsjahr:	2022

In memory of my beloved father
Josef Stotko

In Gedenken an meinen geliebten Vater
Josef Stotko

Acknowledgements

First, I would like to thank Prof. Dr. Reinhard Klein for giving me the chance to work in his group on such exciting problems in the area of geometry and appearance reconstruction and for his continuous engaging and inspiring advices during the last years. Furthermore, I would like to thank Prof. Dr. Matthias B. Hullin for providing many valuable comments during the development of our telepresence application and for serving as a reviewer of this thesis.

I would also like to thank my co-authors (in alphabetical order) Prof. Dr. Sven Behnke, Prof. Dr. Maren Bennewitz, Lukas Bode, Dr. Hassan Errami, Prof. Dr. Juergen Gall, Andreas Görlitz, Prof. Dr. Matthias B. Hullin, Prof. Dr. Reinhard Klein, Prof. Dr. Andreas Kolb, Stefan Krumpen, Oh-Hun Kwon, Christian Lenz, Sebastian Merzbach, Prof. Dr. Matthias Nießner, Radu Alexandru Rosu, Max Schwarz, Julian Tanke, Prof. Dr. Christian Theobalt, Prof. Dr. Andreas Weber, Prof. Dr. Michael Weinmann, Prof. Dr. Angela Yao, Domenic Zingsheim, and Dr. Michael Zollhöfer. Furthermore, I want to thank my colleagues at the computer graphics group for all the helpful comments and discussions regarding my work during the Ph.D. studies. I also would like to thank David Stotko, Domenic Zingsheim, Lukas Bode and Prof. Dr. Reinhard Klein for proof-reading this thesis.

Furthermore, I would like to thank Dr. Richard Newcombe for offering me to work at Facebook Reality Labs as a research intern in 2020 as well as Dr. Vladlen Koltun for offering me an internship at his lab at Intel in 2021.

Finally, I would like to express my deep gratitude to my family and, in particular, to my parents Josef and Maria and to my brother David for their endless support and patience with me during this incredible journey.

Abstract

The recent advances in Augmented Reality (AR) and Virtual Reality (VR) technology and their growing popularity in the past years has significantly influenced emerging trends towards more intuitive and user-centric applications that are accessible to a large community. Sharing immersive experiences is the central challenge here and requires the accurate presentation of virtual content to the user which not only depends on several technical aspects of the respective display devices but also on the visualized scene elements. However, many applications are particularly designed for immediate scenarios which cannot entirely rely on pre-generated content and, thereby, require on-the-fly acquisition of the unknown surrounding scene environment in an efficient and progressive way.

In this thesis, we developed and investigated techniques for real-time reconstruction of 3D scene geometry and appearance with a specific focus on practical methods and systems that could be used in AR and VR scenarios. To this end, we present several contributions which can be categorized into three major areas of these systems. First, we studied the L_1-based Locally Optimal Projection (LOP) operator in the context of data prefiltering and introduced a family of generalized operators in which each element corresponds to a localized L_p estimator. Furthermore, we revealed their close relation to the Mean Shift framework and derived various theoretical properties of the respective kernels and, in turn, the projection operator. We applied the gained insights to define an improved density weighting scheme, a more accurate kernel approximation for the continuous projection operator, as well as a set of robust loss functions which correspond to the kernels. Secondly, we developed a practical multi-client live telepresence system which enables streaming live-captured 3D scene data to remotely connected users who can independently explore and interact with the scene. We introduced a bandwidth-efficient volumetric data structure based on Marching Cubes indices as well as fast GPU hash data structures to efficiently maintain and progressively stream the reconstructed model with only moderate network requirements. In subsequent work, we further improved the overall performance and scalability of our telepresence system by introducing several algorithmic improvements to the reconstruction component. Thirdly, we investigated a segmentation-based approach for estimating appearance information in terms of the spatially-varying surface albedo from RGB-D and additional infrared (IR) input data. In addition to an improved formulation of the coupling between the color and infrared channels, we also incorporated temporal information from previous frames to accelerate the Total Variation-based optimization process.

Contents

Part I

Introduction

Chapter 1

Introduction

The rapid developments of recent Augmented Reality (AR) and Virtual Reality (VR) technology have benefited many applications and are increasingly influencing our everyday lives towards more intuitive and user-centric experiences. Current AR display devices such as the Microsoft HoloLens 2 or the Magic Leap 2 allow to see the physical world with overlaid virtual elements and can, thereby, enhance the visual perception with additional information. On the other hand, VR devices such as the HTC Vive Pro or the Valve Index allow users to completely dive into a virtual world by providing an immersive experience of being and feeling present in there. Natural applications that arise from these capabilities are telepresence and teleconferencing scenarios where people at distant places can connect and interact with each other in a similar way as if they were physically present at a single location. This, in turn, stimulated further applications in other related fields like robotics, architecture, entertainment, education, medicine, and many others.

A crucial component in any of these systems is the accurate presentation of the virtual content to the user which is not only influenced by the actual visualization on the respective AR or VR display devices, determined by technical parameters including framerate, resolution or latency to avoid disturbing effects like motion sickness, but also by the quality and expressiveness of the content itself. Pure image-based data such as 360° images and videos can easily be captured by standard cameras and provide a high amount of details due to the maturity of the camera sensor technology. In this context, the recently presented seminal work in Neural Radiance Fields (NeRF) [Mildenhall et al., 2020] received significant attention as it enabled synthesizing realistic novel unseen views of a scene from an implicit model which was represented by a neural network and optimized only from a set of recorded images and the respective camera poses. A plethora of subsequent work aimed to address several of its initial limitations including to allow representing dynamic scenes [Li et al., 2022], handling varying illumination conditions [Martin-Brualla et al., 2021], reducing the amount of training time [Müller et al., 2022], or increasing its robustness for very sparse sets of input views [Niemeyer et al., 2022]. Nevertheless, many of these approaches still resort to offline optimization and precomputed camera poses, are restricted to small-scale scenes, cannot render high-resolution images in real time due to expensive ray marching, and only have limited interaction and editing functionalities in terms of interpolating between observed states. To this end, traditional concepts from computer graphics and computer vision were leveraged by converting the implicit NeRF representation into classic primitives such as Bidirectional Reflection Distribution Functions (BRDF), environment illumination, and surface geometry in terms of meshes [Zhang et al., 2021b; Yuan et al., 2022] that can be more flexibly edited or exchanged and are also significantly faster to visualize [Chen et al., 2022b]. Similarly, the workflow as well as several components of classic real-time surface reconstruction systems were employed to perform online fusion of radiance fields [Zhang et al., 2022b]. Meeting all of these requirements, i.e. real-time online reconstruction of large-scale scenes *and* real-time live visualization of the dynamically updated model state which is crucial in many VR applications such as telepresence, is, however, still an unsolved problem. Thus, it remains an open question whether such NeRF-based view synthesis approaches will eventually replace traditional reconstruction and rendering concepts, or instead extend them and stimulate further research in these directions [Tewari et al., 2022].

In contrast to these implicit approaches, the direct recovery of explicit surfaces and their appearance properties has already been studied extensively for several decades as one of the fundamental problems in computer vision and computer graphics. Initial work in this area focused on high-fidelity reconstruction of scene geometry from point cloud data which was captured by expensive laser scanning equipment [Levoy et al., 2000]. Besides purely geometric scene information, the estimation of an object's appearance has also been extensively analyzed in the scope of accurately calibrated lab-like setups [Schwartz et al., 2013]. With the recent advances of depth sensing technology and the increasing availability of low-cost commodity hardware such as the Microsoft Kinect and even built-in RGB-D cameras in today's mobile phones, the focus has shifted towards more casual acquisition scenarios and the first classic systems which enable both real-time reconstruction and visualization emerged [Izadi et al., 2011; Newcombe et al., 2011].

In this thesis, we developed methods which contribute towards this goal of real-time live geometry and appearance reconstruction of real-world scenes with commodity RGB-D camera hardware. In particular, we directed our attention onto practical methods to support emerging trends and applications using immersive AR and VR technology.

1.1 Reconstruction Pipeline and Challenges

Significant effort has been invested in advancing the field of real-time 3D reconstruction where many respective approaches follow a common pipeline that can be divided into multiple stages, each describing a distinct sub-problem with several challenges. An overview of this pipeline is shown in Figure 1.1.

Data Prefiltering. The raw input frames captured by low-cost depth sensing devices are of much lower quality than data from more expensive, specialized hardware and exhibit a significantly higher amount of noise. In order to facilitate the processing of the frames in subsequent stages and furthermore increase their reliability, the data is first preprocessed and denoised in an initial step which typically consists of applying a (Gaussian) bilateral

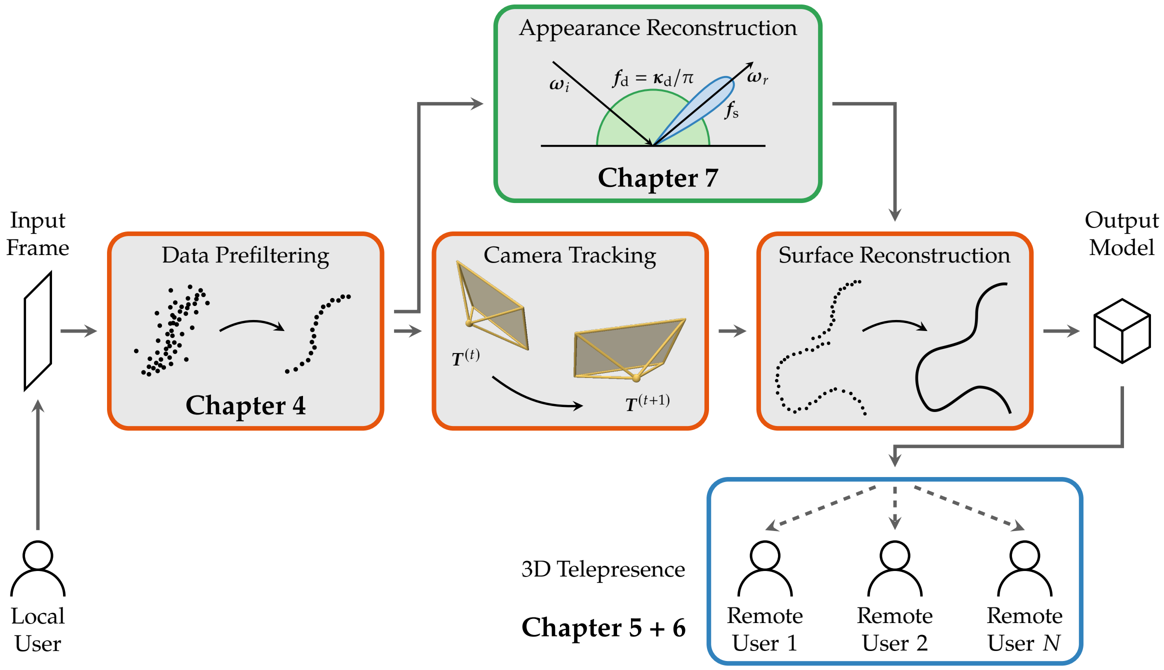


Figure 1.1: Overview of the common stages of 3D reconstruction systems. New input frames captured by the local user are first prefiltered and then used to track the current camera pose and to update the global model in the surface reconstruction step. In addition to the surface geometry, appearance information can be reconstructed and integrated into the 3D model as well. Finally, the captured scene can be progressively streamed to remotely connected users to enable sharing immersive telepresence experiences. Our contributions presented in this thesis are located in data prefiltering, 3D telepresence, and appearance reconstruction.

filter that smooths the samples in the L_2 sense. However, the characteristics of the noise may not necessarily follow a simple Gaussian model, but can instead also include systematic outliers introduced by limitations in the measurement process of the sensor and lead to a distribution that is generally unknown. Furthermore, in case of directly captured point cloud data, the samples are usually not uniformly distributed in 3D space.

Camera Tracking. Since the scene is captured from various angles, the pose of the camera in a canonical world coordinate system should be estimated to bring the acquired data into alignment. This requires finding correspondences between two frames which, however, becomes challenging in scenarios where mostly feature-less regions are visible during the acquisition, such as in the case of moving the sensor in front of a low-textured, planar wall. Fast motions and, in particular, large changes of the camera orientation may introduce motion blur effects in the frames and further increase the difficulty of the correspondence estimation. In addition, the brightness of recorded color images may change rapidly over time in scenes with strongly varying illumination conditions. Although the relative motion to the previous frame is typically small, slight estimation errors in the position and orientation caused by these effects may quickly accumulate and could lead to larger inconsistencies in the model.

Surface Reconstruction. Afterwards, the surface is reconstructed by fusing the captured and aligned frames into a suitable geometric data representation. This process shares many of the challenges of data prefiltering which particularly includes the fact that the input frames can still be noisy and may contain outliers or the fact that the sampling of the surface data is often non-uniform. Furthermore, small misalignments in the camera tracking stage could lead to artifacts and duplicate surface geometry which, however, should be merged into a single surface instead. On the other hand, the surface of thinner objects should be preserved at each side and, in turn, not be accidentally consolidated. Finally, incomplete and missing data poses an additional challenge where small regions in scenes may not have been fully scanned or were partially occluded by other objects but should still be completed.

Appearance Reconstruction. In addition to the aforementioned steps which are related to the geometric aspects of the surface, the appearance can be estimated as well. This is a highly under-constrained problem as the camera only captures the final color information resulting from the complex interplay of light with the geometry and material properties, defined by the diffuse albedo as well as specular components, of the scene. Solving this problem requires incorporating more information about the captured scene in terms of additional data modalities or further priors and assumptions to regularize the formulation. However, overly restrictive constraints or too insufficient regularization can both lead to implausible decomposition results or even introduce artifacts which makes the process of finding a suitable granularity of priors particularly difficult and challenging.

3D Telepresence. In the context of immersive AR and VR applications, further requirements and demands on the performance of the reconstruction and rendering systems are imposed to ensure a pleasant user experience. Providing a high degree of immersion and awareness requires the visualization of scene content at considerably higher resolutions and framerates than on standard displays like monitors to avoid deteriorating effects such as motion sickness. On the other hand, the available network and computing resources at the user site are often highly limited, making systems that run a computationally demanding live-reconstruction of the scene in parallel to high-quality rendering infeasible in such scenarios. Furthermore, the amount of reconstructed and updated 3D data depends on the distance of the visible objects to the camera which may significantly vary over time during acquisition.

1.2 Contributions

In the scope of this thesis, we developed methods to address some of the aforementioned challenges. The key contributions of our work are:

- **Generalized LOP Operators for Filtering.** We reveal the relation of the L_1-based Locally Optimal Projection (LOP) operator to the Mean Shift framework and introduce a novel family of generalized kernels each representing a particular localized L_p estimator. Furthermore, we study the theoretical properties of this kernel family and apply the gained insights in several applications to demonstrate their effectiveness [Stotko et al., 2022].

- **Scalable 3D Telepresence based on Surface Reconstruction.** We present a practical telepresence system for streaming live-captured 3D models in real time to an arbitrary number of remotely connected users which allows them to independently explore the virtual scene. To this end, we introduce a novel bandwidth-efficient voxel data structure for efficient streaming of the 3D data as well as fast GPU hash map and set data structures for maintaining the individual streaming states of the connected clients [Stotko et al., 2019a]. In addition, we propose several algorithmic improvements to the involved reconstruction and server components to further improve the compactness of the reconstructed model and the scalability of the live telepresence system [Stotko et al., 2019d].
- **Segment-wise IR-Guided Albedo Estimation for 3D Reconstruction.** We propose a novel segmentation-based approach for the estimation of spatially-varying albedo information from RGB and IR data captured by hand-held time-of-flight cameras. Furthermore, we improve the performance of the optimization process by incorporating temporal information from previous frames [Stotko et al., 2019e].

1.3 List of Publications

The main parts and contributions of this thesis appeared in the following publications:

- **Patrick Stotko**, Michael Weinmann, and Reinhard Klein.
  "Albedo estimation for real-time 3D reconstruction using RGB-D and IR data."
  ISPRS Journal of Photogrammetry and Remote Sensing (P&RS), 2019.
  DOI: `10.1016/j.isprsjprs.2019.01.018`
- **Patrick Stotko**, Stefan Krumpen, Matthias B. Hullin, Michael Weinmann, and Reinhard Klein.
  "SLAMCast: Large-Scale, Real-Time 3D Reconstruction and Streaming for Immersive Multi-Client Live Telepresence."
  IEEE Transactions on Visualization and Computer Graphics (TVCG), 2019.
  DOI: `10.1109/TVCG.2019.2899231`
- **Patrick Stotko**, Stefan Krumpen, Michael Weinmann, and Reinhard Klein.
  "Efficient 3D Reconstruction and Streaming for Group-Scale Multi-Client Live Telepresence."
  IEEE International Symposium on Mixed and Augmented Reality (ISMAR), 2019.
  DOI: `10.1109/ISMAR.2019.00018`
- **Patrick Stotko**, Michael Weinmann, and Reinhard Klein.
  "Incomplete Gamma Kernels: Generalizing Locally Optimal Projection Operators."
  arXiv:2205.01087 (under review), submitted to IEEE Transactions of Pattern Analysis and Machine Intelligence (TPAMI), 2022.
  DOI: `10.48550/arXiv.2205.01087`

Furthermore, the author contributed to several other publications which, however, are not part of this thesis:

- Michael Zollhöfer, **Patrick Stotko**, Andreas Görlitz, Christian Theobalt, Matthias Nießner, Reinhard Klein, and Andreas Kolb.
  "State of the Art on 3D Reconstruction with RGB-D Cameras."
  Computer Graphics Forum (CGF), 2018.
  DOI: `10.1111/cgf.13386`
- **Patrick Stotko**, Stefan Krumpen, Reinhard Klein, and Michael Weinmann.
  "Towards Scalable Sharing of Immersive Live Telepresence Experiences Beyond Room-scale based on Efficient Real-time 3D Reconstruction and Streaming."
  CVPR Workshop on Computer Vision for AR/VR, 2019.
- **Patrick Stotko**.
  "stdgpu: Efficient STL-like Data Structures on the GPU."
  arXiv:1908.05936, 2019.
  DOI: `10.48550/arXiv.1908.05936`
- 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`
- **Patrick Stotko**, Stefan Krumpen, Max Schwarz, Christian Lenz, Sven Behnke, Reinhard Klein, and Michael Weinmann.
  "A VR System for Immersive Teleoperation and Live Exploration with a Mobile Robot."
  IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2019.
  DOI: `10.1109/IROS40897.2019.8968598`
- Julian Tanke, Oh-Hun Kwon, **Patrick Stotko**, Radu Alexandru Rosu, Michael Weinmann, Hassan Errami, Sven Behnke, Maren Bennewitz, Reinhard Klein, Andreas Weber, Angela Yao, and Juergen Gall.
  "Bonn Activity Maps: Dataset Description."
  arXiv:1912.06354, 2019.
  DOI: `10.48550/arXiv.1912.06354`
- Domenic Zingsheim, **Patrick Stotko**, Stefan Krumpen, Michael Weinmann, and Reinhard Klein.
  "Collaborative VR-based 3D Labeling of Live-captured Scenes by Remote Users."
  IEEE Computer Graphics and Applications (CG&A), 2021.
  DOI: `10.1109/MCG.2021.3082267`

1.4 Thesis Outline

The remainder of this thesis is organized as follows:

Chapter 2. First, we provide an overview of previous developments and the state of the art in the area of surface reconstruction, 3D telepresence systems, and appearance estimation.

Chapter 3. Furthermore, we introduce the relevant theoretical preliminaries and basic concepts for 3D scene reconstruction which will serve as the foundation of the presented methods.

Chapter 4. We present our publication "Incomplete Gamma Kernels: Generalizing Locally Optimal Projection Operators" [Stotko et al., 2022] which already appeared as a preprint and introduces a generalization of Local Optimal Projection operators for point cloud denoising.

Chapter 5. We present a summary of our peer-reviewed publication "SLAMCast: Large-Scale, Real-Time 3D Reconstruction and Streaming for Immersive Multi-Client Live Telepresence" [Stotko et al., 2019a] which introduces a practicable telepresence system for remote collaboration and exploration of live-captured scenes.

Chapter 6. We present a summary of our peer-reviewed publication "Efficient 3D Reconstruction and Streaming for Group-Scale Multi-Client Live Telepresence" [Stotko et al., 2019d] which proposes several algorithmic extensions to the system presented in **Chapter 5** to significantly improve its overall performance and open up further applications in, e.g., education.

Chapter 7. We present a summary of our peer-reviewed publication "Albedo estimation for real-time 3D reconstruction using RGB-D and IR data" [Stotko et al., 2019e] which features a method for reconstructing both the scene geometry and appearance in terms of the spatially-varying albedo in real time.

Chapter 8. Finally, we conclude this thesis by discussing the impact of the presented methods and providing an outlook of potential future directions.

Chapter 2

Related Work

In this chapter, we provide an overview of the literature related to the techniques and systems presented in this thesis. To this end, we first review the initial developments in the field of surface reconstruction (see Section 2.1) as well as advances towards real-time commodity scanning (see Section 2.2) and recent trends in learning-based approaches (see Section 2.3). Furthermore, we discuss the recent achievements in related fields such as the application in 3D telepresence systems (see Section 2.4) or appearance reconstruction (see Section 2.5). A timeline of selected influential and closely related publications is shown in Figure 2.1.

2.1 Classic Surface Reconstruction

Surface reconstruction has been an active research field for several decades where various methods have been presented over time.

Local Point-based Approaches. Early works have been developed starting in the 1990s with the seminal work of Hoppe et al. [1992]. Based on a set of registered point clouds, normals were first computed and then oriented in a consistent manner by propagating normal orientations over neighboring points. In the actual reconstruction step, a signed distance function (SDF) was computed by considering the local tangent plane of the closest point in the input data and then converted to a surface mesh using Marching Cubes [Lorensen and Cline, 1987]. Similar to this tangent plane formulation, *Moving Least Squares* (MLS) approaches [Alexa et al., 2003; Levin, 2004] defined a localized projection operator that first finds the tangent plane that approximates the neighboring points of the target point cloud. Afterwards, the (unsigned) distances of each target point to this plane are used as weights to locally fit a polynomial which represents the MLS surface onto which the query point is then finally projected. The MLS surface could also be defined in a different way via a fixed-point optimization [Amenta and Kil, 2004] or even implicitly in terms of a signed distance function by the Implicit MLS (IMLS) [Kolluri, 2005] and Robust Implicit MLS (RIMLS) [Öztireli et al., 2009] approaches. In contrast to these L_2-based projection operators, the *Locally Optimal Projection* (LOP) operator [Lipman et al., 2007] considered a

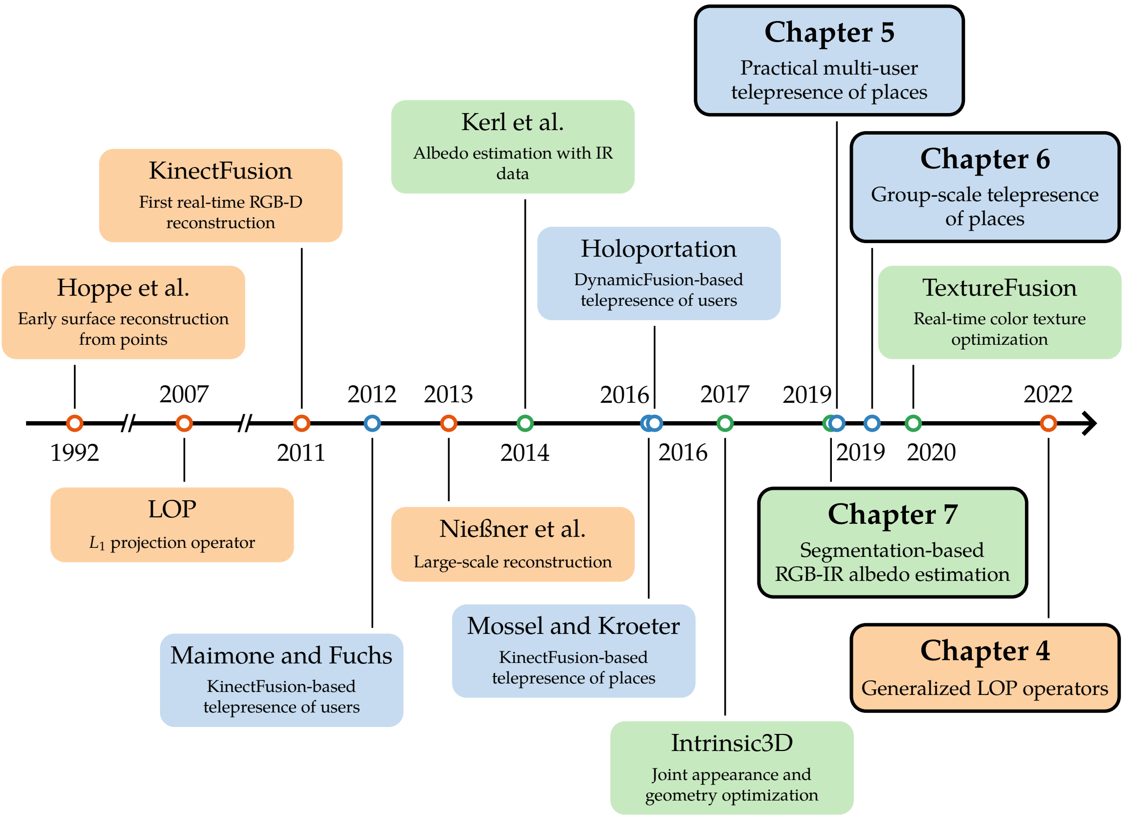

Chapter 5
Practical multi-user telepresence of places
Chapter 6
Group-scale telepresence of places
Kerl et al.
Albedo estimation with IR data
KinectFusion
First real-time RGB-D reconstruction
Holoportation
DynamicFusion-based telepresence of users
Hoppe et al.
Early surface reconstruction from points
TextureFusion
Real-time color texture optimization
2007
2012
2013
2016
2017
2019
2022
1992
2011
2014
2016
2019
2020
LOP
L_1 projection operator
Nießner et al.
Large-scale reconstruction
Chapter 7
Segmentation-based RGB-IR albedo estimation
Mossel and Kroeter
KinectFusion-based telepresence of places
Chapter 4
Generalized LOP operators
Maimone and Fuchs
KinectFusion-based telepresence of users
Intrinsic3D
Joint appearance and geometry optimization

Figure 2.1: Timeline of selected influential and closely related publications as well as their relation to the contributions of this thesis which will be presented in Chapters 4 to 7. Surface reconstruction methods are highlighted in **orange**, telepresence systems in **blue**, and appearance reconstruction approaches in **green**.

L_1-based formulation built upon a localized version of the geometric median which can be computed by, e.g., the Weiszfeld algorithm [Weiszfeld, 1937]. Due to its high robustness in the presence of strong noise and outliers, LOP can be used for point cloud consolidation to obtain a resampled set of clean points which more closely represents the measured surface and which can be passed as the input for other surface reconstruction techniques such as global approaches like Poisson Surface Reconstruction [Kazhdan et al., 2006]. Several extensions of the original LOP approach have been developed to address its limitations including a weighted version to improve the distribution of the projected points [Huang et al., 2009], accelerated variants that either consider subsampling based on kernel density estimation [Liao et al., 2013] or continuous representations [Preiner et al., 2014], as well as anisotropic edge-aware formulations [Huang et al., 2013].

In the scope of our recent work [Stotko et al., 2022] which will be discussed in Chapter 4, we derived a generalization of the LOP operator and took advantage of the gained theoretical insights to further improve on the weighted and continuous versions and to open up other potential applications.

Global Point-based Approaches. Besides the methods that operate locally either in an explicit manner via projection or based on an implicit formulation, global optimization has also been employed for reconstruction from point clouds. Radial basis functions (RBF) were used to define a signed distance function as a weighted sum of the basis which is computed by a global linear optimization of the weights [Carr et al., 2001]. This ensures that the zero-set of the SDF interpolates the observed input points. In more recent work [Liu et al., 2016], Hermite radial basis function (HRBF) implicits have been considered for interpolation and, in the special case of compactly supported kernels [Wendland, 1995], a quasi-solution for interpolating local surface regions can be derived in closed form. *Poisson Surface Reconstruction* (PSR) [Kazhdan et al., 2006] computed a binary indicator function for distinguishing interior and exterior space by casting the respective minimization objective, consisting of the difference between the gradient of the indicator function and a vector field defined by the point cloud normals, to a Poisson problem. In follow-up works, the authors further improved their method by incorporating a screening term to more closely approximate the point set [Kazhdan and Hoppe, 2013] as well as by enforcing envelope constraints to improve the accuracy in regions with missing data [Kazhdan et al., 2020]. Recent work even considered the application of Poisson Surface Reconstruction on unoriented point clouds by iteratively running the method while refining the normal orientations after each reconstruction attempt [Hou et al., 2022]. Other methods considered smooth signed functions instead of indicator functions and constrained the smoothness in the Poisson problem formulation by an additional regularization term based on the Hessian matrix of the SDF [Calakli and Taubin, 2011].

Initial Depth-based Approaches. In contrast to the aforementioned methods which primarily worked on raw point clouds or point clouds with additionally computed normal information, Curless and Levoy [1996] focused on the direct reconstruction from depth image data. While also a signed distance function is used as the underlying representation, their least-squares-based formulation resulted in a simple online update step which particularly allows for an incremental and order-independent processing of the depth data. This approach has been further extended to operate on various discrete hierarchy scales managed by an octree [Fuhrmann and Goesele, 2011] and even to continuous scales based on Gaussian radial basis functions [Fuhrmann and Goesele, 2014].

2.2 Real-time Online Surface Reconstruction

With the more widespread availability of cheap commodity sensors in the 2010s such as the Microsoft Kinect, capturing depth information at high framerates of typically 30 Hz became more popular and the reconstruction of larger scenes with hand-held sensors gained increasing interest.

Volumetric Fusion Methods. The seminal KinectFusion system [Izadi et al., 2011; Newcombe et al., 2011] marked the cornerstone of this trend as it was the first system that enabled the reconstruction of room-sized scenes with interactive feedback to the user by running not only camera tracking but also dense surface reconstruction in real time. Key to their success was a fully GPU-accelerated pipeline consisting of an efficient projection-based depth image registration algorithm [Rusinkiewicz et al., 2002], which aligned new input data against an on-the-fly generated depth image of the current 3D model resulting in a significantly improved tracking accuracy, as well as the fast incremental integration of these input data into a volumetric signed distance function representation [Curless and Levoy, 1996]. In the following years, several extensions have been developed to address the limitations of the KinectFusion system. Whereas initially only a fixed-sized cubic volume consisting of typically 512 voxels per dimension was used limiting the size of the reconstructed model, moving volume techniques [Roth and Vona, 2012; Whelan et al., 2012; Whelan et al., 2015a] as well as hierarchical data structures [Zeng et al., 2012; Chen et al., 2013; Steinbrücker et al., 2013], spatially-hashed data structures [Nießner et al., 2013; Kähler et al., 2015] or even a combination of the latter two [Kähler et al., 2016b] have been proposed to increase the storage efficiency of the volumetric representation and to enable the live reconstruction of large-scale scenes of theoretically almost unrestricted size. Besides these improvements to the internal data representation, the camera tracking component also received significant attention by considering loop closure techniques that build a graph of nearby camera poses and perform a global optimization of all poses. This includes approaches that divided the scene into individually reconstructed small overlapping submaps which were continuously aligned by the pose graph [Kähler et al., 2016a], as well as keyframe-based approaches which directly performed a correction of previous input data via on-the-fly de-integration and re-integration into the 3D model [Dai et al., 2017]. Further work considered collaborative scenarios where multiple users, each equipped with a RGB-D camera, independently scan parts of a large scene, such as a building, to obtain a single merged 3D model [Golodetz et al., 2018], or even scenarios where the scene is scanned in a fully autonomous manner by multiple moving robots [Dong et al., 2019]. Finally, instead of relying on GPU-accelerated processing of the input data which requires powerful hardware, approaches that run purely on the CPU [Steinbrücker et al., 2014; Han and Fang, 2018] have been developed including methods that are capable of running on mobile devices [Klingensmith et al., 2015].

Surfel Fusion Methods. Whereas KinectFusion and all related approaches that we discussed so far were based on a volumetric grid and, thereby, used an implicit surface representation, the advantages of other data structures such as surfels [Pfister et al., 2000] as the underlying representation were also leveraged. Stückler and Behnke [2014] used a hybrid representation where surfels generated from the input depth images are stored and integrated in a hierarchical octree data structure to ensure a regular distribution of the surfels. A flexible method inspired by the KinectFusion system has been presented by Keller et al. [2013] who followed the same general pipeline, but applied the volumetric update step [Curless and Levoy, 1996] directly on the explicit surfel data to incrementally build a large surfel model that is more compact than implicit voxel data structures. Similar to the discussed developments in overcoming the limitations of KinectFusion, several extensions to the surfel-based fusion approach were presented which particularly included improved camera tracking using loop closure [Whelan et al., 2015b] or direct bundle adjustment [Schöps et al., 2019a], as well as modeling anisotropic noise in the fusion process [Lefloch et al., 2015] and in the tracking component [Cao et al., 2018] to improve the accuracy of the reconstructed models. Recently, closed-form interpolation with Hermite radial basis function implicits [Liu et al., 2016] has been applied in fast surfel-based fusion to continuously evaluate the surface and, thereby, increase the tracking and reconstruction accuracy [Xu et al., 2022b].

2.3 Learning-based Surface Reconstruction

Due to the huge success of modern deep learning approaches in various computer vision tasks, much work has been invested into learning-based surface reconstruction from point cloud or depth data.

Offline Learning. Early methods constructed an atlas of learned local surface parametrizations and obtained the final surface via composition of the local patches [Groueix et al., 2018; Williams et al., 2019]. Implicit functions also became very popular as the underlying data representation. Riegler et al. [2017] used a 3D convolutional neural network (CNN) architecture to learn the signed distance function using an octree representation from a set of depth images. In order to avoid the limitations of discrete voxel representations, the seminal works of Park et al. [2019] and Mescheder et al. [2019] proposed to use fully connected networks, i.e. multi-layer perceptrons (MLP), for learning continuous signed distance functions or continuous occupancy fields respectively. Recent approaches followed this trend and learned a continuous SDF from a sequence of RGB-D images [Azinović et al., 2022] or from point clouds [Ma et al., 2021]. Some techniques also applied ideas from classic surface reconstruction (see Section 2.1) by incorporating priors based on Implicit Moving Least Squares [Liu et al., 2021b; Wang et al., 2021] or a differentiable formulation of Poisson Surface Reconstruction [Peng et al., 2021].

Online Learning. Since the aforementioned offline approaches require significant computing resources as well as repeated access to the whole fully captured dataset, augmenting classic real-time online approaches (see Section 2.2) with neural networks recently gained increasing attention. Instead of applying the classic volumetric update step [Curless and Levoy, 1996] which effectively results in a local average, some methods learned a non-linear SDF update function to improve the reconstruction quality [Weder et al., 2020], or performed data fusion in a latent space where the fused SDF can be obtained from the latent representation by an additional translation network [Rückert and Stamminger, 2021; Weder et al., 2021]. A similar idea has been applied for online fusion of point cloud data in a latent space [Lionar et al., 2021]. In the context of collaborate scanning, recent work demonstrated how to learn the noise characteristics of different sensors to perform multi-sensor data fusion [Sandström et al., 2022]. Finally, continual learning of the signed distance function has also been considered for online data fusion where samples of previous network predictions of the SDF are stored in a fixed-size buffer and replayed in future updates to maintain these learned structures within the network [Yan et al., 2021; Ortiz et al., 2022]. Despite these advances in learning-based fusion, current real-time methods are either inherently limited in terms of reconstruction resolution or achieve this performance at higher resolutions only when considering a small subset of keyframes.

2.4 3D Telepresence Systems

Similar to its impact on the recent trends in the field of surface reconstruction, the release of cheap RGB-D cameras like the Microsoft Kinect also opened up new possibilities for telepresence applications.

Frame-based Data Fusion. In initial systems, multiple Kinect cameras were placed in a room to capture the interior from various angles and a visibility-based method was proposed to merge the overlapping depth images [Maimone and Fuchs, 2011; Maimone et al., 2012]. With the great success of the KinectFusion system [Izadi et al., 2011; Newcombe et al., 2011] for rapid scene capturing and reconstruction, 3D telepresence systems improved significantly in terms of visualization quality. Instead of only relying on the RGB-D images of multiple Kinect sensors from a single time step, Maimone and Fuchs [2012] applied KinectFusion to dynamic scenes and modified the volumetric update step to quickly replace detected persons and other moving parts within the scene with new measurements while keeping the static parts of the scene. Further work considered a two-stage capturing process where initially a 3D model of the static and semi-static parts of the scene is reconstructed with a single moving RGB-D camera and subsequently overlaid with the dynamic scene content captured from various fixed RGB-D cameras [Dou and Fuchs, 2014]. The recent Starline project [Lawrence et al., 2021] demonstrated a high-fidelity telepresence system which, in addition to replacing volumetric data fusion with faster image-based fusion, also accounted for spatialized audio using head tracking and, thereby, provided a much higher degree of immersion than 2D teleconferencing systems.

Temporal Data Fusion. Although the limitations of handling dynamic scene content in the KinectFusion system, which was originally designed for static scenes, have been partially mitigated, fully consistent integration was demonstrated only later by DynamicFusion [Newcombe et al., 2015] which performed non-rigid tracking in real time by estimating a dense volumetric warp field that represents the individual motion of each point in space. Successive methods not only focused on a more accurate tracking in general, but also considered multiple RGB-D cameras to handle fast motions [Dou et al., 2016]. Based on these impressive developments, the Holoportation system [Orts-Escolano et al., 2016] demonstrated to provide an immersive telepresence experience using Augmented Reality devices such as the Microsoft HoloLens by performing incremental reconstruction and streaming of fully dynamic 3D models of the person and their surrounding objects to another user in real time.

Scene-oriented Telepresence. The primary focus of the aforementioned systems lied onto 3D telepresence of users and, thereby, onto scenarios which are limited to small and controlled room-like scenes. However, telepresence of places that directly targets the surrounding scene environment also received increasing attention in recent years. Mossel and Kroeter [2016] developed a system that performed large-scale scene reconstruction using an efficient spatially-hashed extension of KinectFusion [Kähler et al., 2015] and simultaneously streamed the internal volumetric 3D model representation to another remotely connected user over time. This way, the remote user was able to virtually explore the scene during acquisition using Virtual Reality devices. In the context of Mixed Reality (MR), such systems were extended to allow for closer collaboration [Zillner et al., 2018] where the local user captures a reconstructed 3D model of the environment via an AR device with an integrated RGB-D camera and sees the interactions of a remote user with the virtual model. Other approaches captured the environment with a calibrated cluster of RGB-D cameras and streamed the stitched 3D panorama point clouds to a VR remote user [Sasikumar et al., 2019; Bai et al., 2020]. Similarly, the MobilePortation system [Young et al., 2020] allowed to incrementally capture a 3D point cloud model of the scene using mobile phones with integrated RGB-D cameras by filtering duplicate points with a simple voxel grid filter. To increase the degree of immersion, a live 360° video was streamed in parallel and blended with the 3D model when the remote user approached the local user's position.

In the scope of our publications introducing SLAMCast [Stotko et al., 2019a; Stotko et al., 2019d] which will be discussed in Chapters 5 and 6, we developed a practical telepresence system which is not limited to single-user data streaming, but instead allows to share live-reconstructed, large-scale 3D models of the surrounding scene environment to multiple remotely connected users by introducing a bandwidth-efficient data structure for streaming and managing the respective client states with fast GPU hash map data structures.

2.5 Appearance Reconstruction

Besides the reconstruction of the shape of objects in terms of surface geometry, their appearance also plays an important role in the creation of high-quality virtual models and is captured by the complex interplay between material properties such as the surface albedo, the surrounding illumination conditions as well as the viewing direction. Many methods that are built upon the volumetric update step of KinectFusion [Izadi et al., 2011; Newcombe et al., 2011] apply a similar weighted averaging scheme to estimate surface color information which, however, bakes all illumination-dependent effects into the reconstructed texture.

Texture Optimization. In order to improve the texture quality, highlight detection was used to remove samples from the integration process and, thereby, from the final texture [Whelan et al., 2016]. Since RGB-D cameras automatically adjust the exposure time of the captured RGB images to ensure that the observed dynamic range of intensities can be well represented by the limited capabilities of the sensor, other approaches proposed to estimate this exposure time to obtain a consistent high-dynamic-range color texture instead of directly fusing the raw low-dynamic-range images [Li et al., 2016]. More sophisticated methods formulated the problem as a large offline optimization objective to increase the overall sharpness of the reconstructed texture and the amount of reconstructed details [Zhou and Koltun, 2014; Bi et al., 2017; Fu et al., 2021]. Recent work also considered online texture fusion in real time by storing texture tiles per voxel [Lee et al., 2020; Kim et al., 2022], as well as learning-based texture optimization using adversarial formulations [Oechsle et al., 2019; Huang et al., 2020b] or differentiable rendering [Dai et al., 2021].

Albedo Estimation. Although texture optimization improves the quality and consistency of the object colors, the surrounding illumination is still inherently tied into the texture which makes these approaches unsuitable in various scenarios such as relighting applications where a reconstructed object is placed into another scene with a different illumination, e.g. in the context of Augmented or Mixed Reality. Separating the shading to recover the surface material in terms of the diffuse albedo has been formulated as the ill-posed intrinsic image decomposition problem where a broader overview of various classic and learning-based approaches can be found in related surveys [Bonneel et al., 2017; Garces et al., 2022]. Popular choices to address the inherent scale ambiguity of the intrinsics image decomposition problem include smoothness priors for the albedo and shading images based on L_1 regularization of the gradients [Kerl et al., 2014] or L_2 regularization respectively [Meka et al., 2016]. Several approaches used depth data to define more expressive priors for guiding the optimization process [Barron and Malik, 2013; Chen and Koltun, 2013; Hachama et al., 2015]. Since the scene is actively illuminated with infrared (IR) light by time-of-flight cameras in the depth measurement process, Kerl et al. [2014] estimated the respective IR albedo from this controlled illumination setup and applied it in an image-wide coupling term to resolve the ambiguity. Recent work also considered natural infrared illumination [Cheng et al., 2019b] or even hyperspectral images [Zhang et al., 2022a] to regularize the gradients of the decomposed shading images. Furthermore, additional constraints to enforce piecewise constant albedo or shading values within clusters were introduced based on image segmentation [Shi et al., 2015] or from user inputs in interactive scene editing applications [Meka et al., 2017]. The latter approach has been recently extended in a global-illumination-based formulation by decomposing the shading image into a linear combination of indirect illumination layers that model inter-reflections with respect to estimated base color clusters [Meka et al., 2021].

In the scope of our publication [Stotko et al., 2019e] which will be discussed in Chapter 7, we demonstrated accelerated albedo estimation from RGB-D and IR data within common real-time surface reconstruction systems and introduced a segmentation-based approach to improve the coupling between the RGB and IR albedo images.

BRDF Estimation. Since the diffuse albedo only partially captures the material properties and the appearance of objects, some methods considered the estimation of Bidirectional Reflection Distribution Functions (BRDF) as a more sophisticated model to also faithfully handle specular effects. Wu and Zhou [2015] proposed a multi-stage capturing process where the geometry is first estimated by the KinectFusion system and subsequently used to estimate the diffuse surface albedo as well as the specular component in terms of the

Ward model [Ward, 1992]. Similar multi-stage approaches were developed in the context of Mixed Reality for relighting applications [Richter-Trummer et al., 2016]. Due to the higher resolution and lower amount of noise in RGB images in comparison to depth data, joint appearance estimation and surface geometry refinement was explored to obtain higher quality reconstruction results [Wu et al., 2016; Maier et al., 2017a]. Recent learning-based approaches allowed to estimate material properties as well as the environment illumination of a scene based on differentiable path tracing [Azinović et al., 2019] or even in image space from a single image [Li et al., 2020b].

Chapter 3

Background

In this chapter, we will introduce the basic concepts and tools for 3D scene reconstruction. After introducing the theoretical foundations of energy optimization (see Section 3.1), we will discuss the data acquisition process (see Section 3.2) as well as the initial data prefiltering (see Section 3.3) and camera tracking stages (see Section 3.4). Furthermore, we will review common geometric data representations (see Section 3.5) and describe the steps in volumetric surface reconstruction (see Section 3.6) in more detail. Finally, we will also discuss the preliminaries of image-based appearance reconstruction in terms of intrinsic image decomposition (see Section 3.7).

3.1 Energy Optimization

A common tool to obtain suitable solutions for various problems in the field of computer graphics and computer vision is the optimization of an energy function. In its most general form

$$\boldsymbol{\theta}^* = \operatorname*{arg\,min}_{\boldsymbol{\theta} \in \mathbb{R}^m} E(\boldsymbol{\theta}) \tag{3.1}$$

we are interested in finding the optimal parameters $\boldsymbol{\theta}^* \in \mathbb{R}^m$ which minimize an arbitrary energy function $E(\boldsymbol{\theta})$.

3.1.1 Linear Least Squares Optimization

A very popular choice for the energy function is the *Least Squares* regression model. Given a set of n measurements, each of them is associated with a residual function $r_i \colon \mathbb{R}^m \to \mathbb{R}$ and the contributions of each residual are consolidated into a sum which leads to the following formulation of the energy function:

$$E(\boldsymbol{\theta}) = \sum_{i=1}^{n} r_i(\boldsymbol{\theta})^2 = \|\boldsymbol{r}(\boldsymbol{\theta})\|_2^2 \tag{3.2}$$

Here, the energy can be rewritten in a simpler form by introducing a vector-valued function $\boldsymbol{r}\colon \mathbb{R}^m \to \mathbb{R}^n$ that includes the individual residual terms component-wise. Whereas the residual terms r_i can be non-linear in general, linear functions are often employed as a simple model where the energy function reduces to a linear system of equations $E(\boldsymbol{\theta}) = \|\boldsymbol{A}\,\boldsymbol{\theta} - \boldsymbol{b}\|_2^2$ and the solution $\boldsymbol{\theta}^* = (\boldsymbol{A}^\mathsf{T}\,\boldsymbol{A})^{-1}\boldsymbol{A}^\mathsf{T}\,\boldsymbol{b}$ can be efficiently obtained by the well-known normal equations.

For example, the weighted distance $r_i(\boldsymbol{y}) = \sqrt{w_i}\,\|\boldsymbol{y} - \boldsymbol{x}_i\|_2$ is chosen in various problem formulations to define the energy function

$$E(\boldsymbol{y}) = \sum_{i=1}^{n} w_i\,\|\boldsymbol{y} - \boldsymbol{x}_i\|_2^2 \tag{3.3}$$

where the objective is to find a vector $\boldsymbol{y} \in \mathbb{R}^d$ which best approximates a set of potentially noisy samples $\boldsymbol{x}_i \in \mathbb{R}^d$ associated with accompanying weights $w_i \in \mathbb{R}_{>0}$ in the least squares sense. The solution of this problem is the well-known weighted mean

$$\boldsymbol{y}^* = \frac{\sum_{i=1}^{n} w_i\,\boldsymbol{x}_i}{\sum_{i=1}^{n} w_i} \tag{3.4}$$

which is one of several possible formulations to quantify the *central tendency* of the samples.

3.1.2 Geometric Median

Another possibility to define the central tendency is the *geometric median* which can be considered as a special case of the more general *Least Absolute Deviation* regression model that is defined by the energy function

$$E(\boldsymbol{\theta}) = \sum_{i=1}^{n} |r_i(\boldsymbol{\theta})| = \|\boldsymbol{r}(\boldsymbol{\theta})\|_1 \tag{3.5}$$

and consolidates the individual residual terms in an L_1 sense. Similar to the aforementioned example, by choosing the residual function $r_i(\boldsymbol{y}) = w_i\,\|\boldsymbol{y} - \boldsymbol{x}_i\|_2$, the geometric median of a set of samples $\boldsymbol{x}_i \in \mathbb{R}^d$ with associated weights $w_i \in \mathbb{R}_{>0}$ defines the point in space that minimizes the weighted absolute distances to the samples:

$$E(\boldsymbol{y}) = \sum_{i=1}^{n} w_i\,\|\boldsymbol{y} - \boldsymbol{x}_i\|_2 \tag{3.6}$$

Since no analytic closed-form solution to this problem exists, the value can only be computed numerically via iterative methods such as the Weiszfeld algorithm [Weiszfeld, 1937] which can be derived by considering the partial derivative

$$\mathbf{0} = \frac{\partial}{\partial \boldsymbol{y}} E(\boldsymbol{y}) = \sum_{i=1}^{n} w_i \, \frac{\boldsymbol{y} - \boldsymbol{x}_i}{\|\boldsymbol{y} - \boldsymbol{x}_i\|_2} \tag{3.7}$$

of the energy. After rearranging the terms, the resulting equation reveals a recursive dependency of the optimal solution $\boldsymbol{y}^*$ with itself and, at the same time, defines the corresponding fixed-point iteration step

$$\boldsymbol{y}^{(k+1)} = \frac{\sum_{i=1}^{n} \widetilde{w}_i(\boldsymbol{y}^{(k)}) \, \boldsymbol{x}_i}{\sum_{i=1}^{n} \widetilde{w}_i(\boldsymbol{y}^{(k)})}, \qquad \widetilde{w}_i(\boldsymbol{y}) = \frac{w_i}{\|\boldsymbol{y} - \boldsymbol{x}_i\|_2} \tag{3.8}$$

of the Weiszfeld algorithm. This iterative process can in fact be viewed as repeatably computing the weighted mean with dynamically changing weights with respect to the current estimate $\boldsymbol{y}^{(k)}$ and, thereby, also be reformulated in terms of the following *Iteratively Reweighted Least Squares* (IRLS) problem

$$\begin{aligned} E(\boldsymbol{y}) &= \sum_{i=1}^{n} w_i \, \|\boldsymbol{y} - \boldsymbol{x}_i\|_2 \\ &= \sum_{i=1}^{n} \frac{w_i}{\|\boldsymbol{y} - \boldsymbol{x}_i\|_2} \, \|\boldsymbol{y} - \boldsymbol{x}_i\|_2^2 \\ &\approx \sum_{i=1}^{n} \widetilde{w}_i(\boldsymbol{y}^{(k)}) \, \|\boldsymbol{y} - \boldsymbol{x}_i\|_2^2 \end{aligned} \tag{3.9}$$

with the same weights $\widetilde{w}_i(\boldsymbol{y}^{(k)})$.

3.1.3 Total Variation Regularization

Whereas the aforementioned formulations effectively describe data terms which are designed for finding parameters $\boldsymbol{\theta} \in \mathbb{R}^m$ that best approximate a set of given samples, *Total Variation* (TV) regularization instead imposes additional smoothness constraints on the parameter coordinates. In particular, we now consider the more general case of finding a function $\boldsymbol{f} \colon \Omega \longrightarrow \mathbb{R}^m$ parameterized over a l-dimensional domain $\Omega \subseteq \mathbb{R}^l$ instead of a single vector $\boldsymbol{\theta}$. The set of functions $\boldsymbol{f}$ forms a vector space and can be associated with the (canonical) inner product and a p-norm that are defined in a similar way as for finite-dimensional vectors by

$$\langle \boldsymbol{f} | \boldsymbol{g} \rangle := \int_{\Omega} \langle \boldsymbol{f}(\boldsymbol{x}) | \boldsymbol{g}(\boldsymbol{x}) \rangle \, \mathrm{d}\boldsymbol{x}, \qquad \|\boldsymbol{f}\|_p := \left[\int_{\Omega} |\boldsymbol{f}(\boldsymbol{x})|^p \, \mathrm{d}\boldsymbol{x} \right]^{1/p} \tag{3.10}$$

where $|\boldsymbol{f}(\boldsymbol{x})|$ denotes the euclidean length, i.e. $|\boldsymbol{f}(\boldsymbol{x})| := \|\boldsymbol{f}(\boldsymbol{x})\|_2$. Since the domain Ω is a non-finite set in general, we require the functions to be square-integrable to ensure a well-defined inner product and, in addition, p-times integrable for the p-norm. Furthermore, the functions should be at least once continuously differentiable in order to compute gradients $\nabla \boldsymbol{f} \colon \Omega \to \mathbb{R}^{m \times l}$.

Based on this definition, a regularized energy function taking a function $\boldsymbol{f}$ as argument can then be defined as

$$\begin{aligned} E(\boldsymbol{f}) &= E_{\text{data}}(\boldsymbol{f}) + \int_{\Omega} |\nabla \boldsymbol{f}(\boldsymbol{x})| \,\mathrm{d}\boldsymbol{x} \\ &= E_{\text{data}}(\boldsymbol{f}) + \|\nabla \boldsymbol{f}\|_1 \end{aligned} \tag{3.11}$$

and consists of a data term E_{data}, which can be formulated, e.g., in the L_2 sense by a linear least squares term or by other means, as well as the Total Variation regularization term, which penalizes local deviations of $\boldsymbol{f}$ within the domain Ω in the L_1 sense. In order to control the influence of the regularization, an additional balancing parameter $\lambda \in \mathbb{R}_{>0}$ can be introduced within the data term E_{data} as a trade-off between the smoothness of the function $\boldsymbol{f}$ and its fidelity. Several algorithms for solving the above regularized optimization problem have been proposed including the popular primal-dual approach by Chambolle and Pock [2011] where the original minimization task is reformulated to the saddle point problem

$$E(\boldsymbol{f}, \boldsymbol{G}) = E_{\text{data}}(\boldsymbol{f}) + \langle \nabla \boldsymbol{f} | \boldsymbol{G} \rangle - \delta_1(\boldsymbol{G}) \tag{3.12}$$

in which the primal variable $\boldsymbol{f}$ is minimized and the respective dual variable $\boldsymbol{G}$ is maximized. Here, the Dirac delta function

$$\delta_1(\boldsymbol{G}) = \begin{cases} 0 & \text{if } \|\boldsymbol{G}\|_\infty \leq 1 \\ +\infty & \text{otherwise} \end{cases} \tag{3.13}$$

restricts the solution space of the dual variable to functions with bounded maximum norm. Furthermore, the primal and dual variables are connected through the gradient operator ∇ and its adjoint version $\nabla^\mathsf{T} = -\operatorname{div}$ which allows to map between the two function spaces.

Given an initialization of the primal and dual variables $\boldsymbol{f}^{(0)}$ and $\boldsymbol{G}^{(0)}$ along with the auxiliary variable $\bar{\boldsymbol{f}}^{(0)} = \boldsymbol{f}^{(0)}$ as well as the step sizes $\sigma, \tau \in \mathbb{R}_{>0}$ and $\theta \in [0, 1]$, the primal-dual solver iteratively applies the following steps:

1. Gradient ascent step: $\boldsymbol{G}^{(k+1)} = \operatorname{prox}_{\sigma\, \delta_1}(\boldsymbol{G}^{(k)} + \sigma\, \nabla \bar{\boldsymbol{f}}^{(k)})$
2. Gradient descent step: $\boldsymbol{f}^{(k+1)} = \operatorname{prox}_{\tau\, E_{\text{data}}}(\boldsymbol{f}^{(k)} - \tau\, \nabla^\mathsf{T} \boldsymbol{G}^{(k+1)})$
3. Extrapolation step: $\bar{\boldsymbol{f}}^{(k+1)} = \boldsymbol{f}^{(k+1)} + \theta\, (\boldsymbol{f}^{(k+1)} - \boldsymbol{f}^{(k)})$

The involved proximal operator

$$\operatorname{prox}_{\tau\, E_{\text{data}}}(\boldsymbol{f}) = \arg\min_{\boldsymbol{g}} \frac{\|\boldsymbol{f} - \boldsymbol{g}\|_2^2}{2\tau} + E_{\text{data}}(\boldsymbol{g}) \tag{3.14}$$

can be interpreted as a damped minimizer of the considered function, which is E_{data} in this case, where the solution is additionally constrained to lie close to the specified value. In case of the dual variable, the proximal operator reduces to a projection to the solution space:

$$\left[\operatorname{prox}_{\sigma\,\delta_1}(\boldsymbol{G})\right](\boldsymbol{x}) = \frac{\boldsymbol{G}(\boldsymbol{x})}{\max(1, |\boldsymbol{G}(\boldsymbol{x})|)} \tag{3.15}$$

We will built upon this formulation and the respective primal-dual solver in our albedo estimation approach which will be described in Chapter 7.

3.2 Data Acquisition

Since the focus of this thesis lies on the reconstruction of indoor scenes, we will primarily consider approaches that operate on RGB-D image frames and are explicitly designed to take advantage of this structured data format (see Figure 3.1). However, we also studied more general approaches that only require unstructured data in terms of 3D point clouds (see Chapter 4). In the following, we will thus also consider point-based sensing devices in addition to purely imaged-based ones.

3.2.1 Depth Sensing

Several types of cameras have been developed for measuring depth information which can be divided into two major categories: *passive capturing* which includes stereo cameras and *active capturing* which is employed by structured light and time-of-flight cameras.

Stereo Cameras. In contrast to active capturing devices which emit a signal into the scene and capture the reflected response, stereo cameras are passive sensors consisting of a pair of RGB cameras that record the scene from slightly different angles. By finding correspondences between the two captured images, the depth of the observed 3D point can be estimated using triangulation and the known fixed baseline between the cameras. While stereo cameras can be used both in indoor and outdoor scenes, the accuracy of the depth image heavily depends on the reliability of the estimated correspondences which becomes challenging for low-textured objects.

Structured Light Cameras. Structured light cameras also rely on correspondence estimation, but consider a more controlled setup where measurements are taken in the infrared (IR) spectrum instead of the visible spectrum. In particular, an IR emitter projects a known structured pattern, e.g. a speckle pattern in case of the Microsoft Kinect v1, into the scene which is then observed by an IR camera. Depth estimation follows the same principle as in the case of stereo cameras by performing triangulation on the matched correspondences which, however, is a significantly easier detection task in this scenario due to the known emitter pattern. Therefore, structured light cameras are well-suited for capturing near objects in indoor scenes within a range of up to five meters but become less reliable in

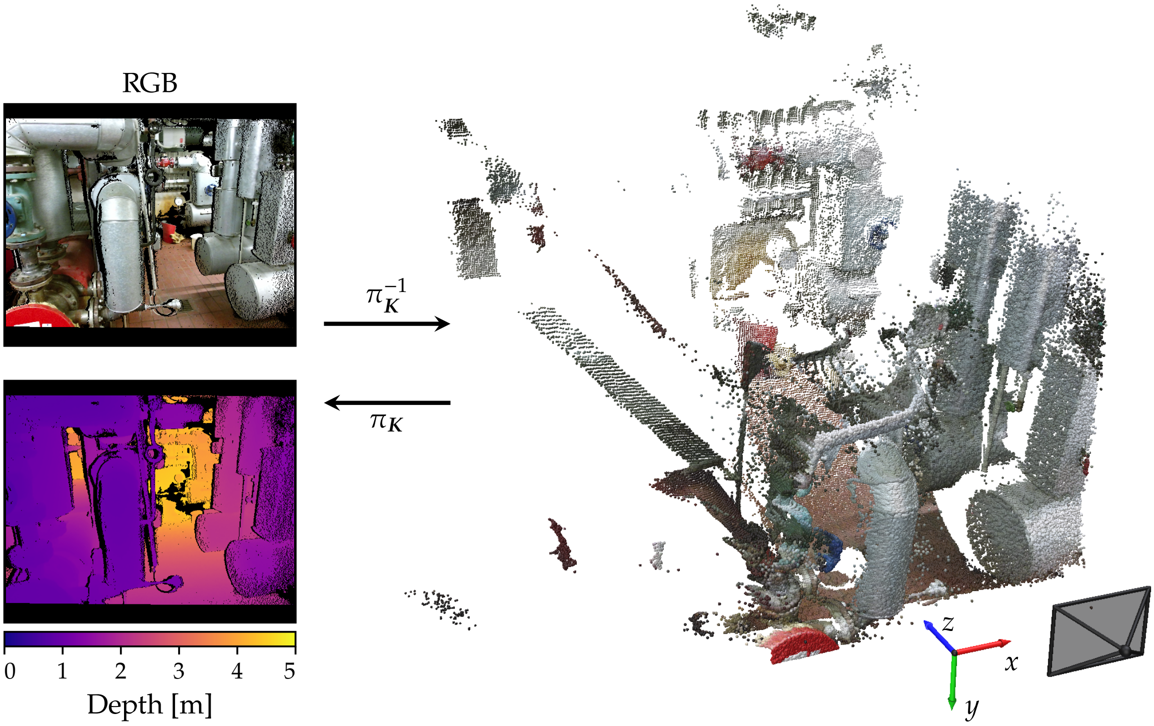


Figure 3.1: Exemplary RGB-D frame from the *heating_room* dataset [Stotko et al., 2019a]. A colored 3D point cloud can be computed via back-projection of the depth image.

outdoor scenarios where the emitted signal may not be distinguished from the surrounding environment illumination, e.g. direct sunlight, anymore.

Time-of-Flight Cameras. Another increasingly popular depth sensing technique is based on time-of-flight imaging which can be categorized into pulsed and continuous wave modulation approaches [Horaud et al., 2016]. Indoor scanning devices such as the Microsoft Kinect v2 usually emit modulated infrared light with a fixed known frequency and measure the phase shift of the reflected signal per pixel. Since the speed of light is a known constant, the overall traveled distances of the light can be computed from the pixel-wise measured phase shifts and then converted to respective depth values by taking half of the distances. Similarly, pulsed time-of-flight cameras emit short pulses of infrared light into the scene and directly measure the time until the pulse is reflected back to the device. While the underlying assumption of a single direct reflection back to the sensor usually holds for most parts of the scene, the emitted light can be reflected more than once around corners or concave edges which results in multiple measurements and can lead to systematically overestimated depth values.

LiDAR Sensors. A subclass of the general set of time-of-flight devices are light detection and ranging (LiDAR) sensors which utilize laser beams to emit light into the scene. Since these sensors are usually equipped with a single or a very low number of laser beams and each of them can only contribute to a single depth measurement, a rotating mirror is used to vary the directions of the laser beams and, thereby, to progressively scan the scene within a frame. Therefore, the measured output provided by LiDAR devices typically consists of 3D point clouds whereas time-of-flight cameras like the Microsoft Kinect v2 capture 2D depth images which can then be projected back to 3D space using the pinhole camera model. Furthermore, LiDAR sensors are often employed in outdoor scenes where measuring larger distances with high accuracy is crucial which includes various scenarios such as autonomous driving [Li and Ibanez-Guzman, 2020].

3.2.2 Pinhole Camera Model

While the captured RGB and depth information is provided in 2D image space, the following processing steps operate on scene information that is given in 3D space. This, in turn, requires the capability to convert between the depth values p_z measured at 2D pixel coordinates $\boldsymbol{u} = (u_x, u_y)^\top \in \mathbb{R}^2$ and the corresponding 3D points $\boldsymbol{p} = (p_x, p_y, p_z)^\top \in \mathbb{R}^3$ which can be defined in terms of perspective projection and back-projection functions. To this end, the pinhole camera model is employed which is a simple model that assumes lens-free projection along a ray to a single point and can be described by the camera calibration matrix

$$\boldsymbol{K} = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \tag{3.16}$$

consisting of the focal lengths $(f_x, f_y)^\top \in \mathbb{R}^2$ of the camera and the principal point $(c_x, c_y)^\top \in \mathbb{R}^2$ which depicts the center of projection in image space. These device-specific properties of the camera can be estimated in a calibration process where a series of images of a known target object, e.g. a checkerboard pattern or another suitable reference object, are taken [Zhang, 2000]. Since the pinhole model does not explicitly handle lens distortions which usually cannot be neglected for actual real-world cameras, the respective distortion parameters are estimated as well during the initial calibration and used in the prefiltering step (see Section 3.3) to compute an undistorted image from the raw sensor data. Based on this camera matrix $\boldsymbol{K}$, the perspective projection $\pi_{\boldsymbol{K}}$ of a 3D point to an undistorted 2D pixel is defined in vector notation by $(\boldsymbol{u}, 1)^\top = \boldsymbol{K} \cdot \boldsymbol{p} / p_z$. Ignoring the fixed last coordinate, we can also write this expression more compactly as:

$$\begin{aligned} \pi_{\boldsymbol{K}} \colon \mathbb{R}^3 &\to \mathbb{R}^2 \\ \boldsymbol{p} &\mapsto \begin{pmatrix} f_x \, \dfrac{p_x}{p_z} + c_x & f_x \, \dfrac{p_y}{p_z} + c_y \end{pmatrix}^\top \end{aligned} \tag{3.17}$$

Similarly, the inverse of the projection $\pi_{\boldsymbol{K}}^{-1}$, i.e. the back-projection of a 2D pixel with a given depth value $p_z = z(\boldsymbol{u})$ into 3D space, is defined as $\boldsymbol{p} = \boldsymbol{K}^{-1} \cdot (\boldsymbol{u}, 1)^\top \cdot p_z$ and can also be rewritten in a more compact form as:

$$\begin{aligned} \pi_{\boldsymbol{K}}^{-1} &: \mathbb{R}^2 \times \mathbb{R} \to \mathbb{R}^3 \\ (\boldsymbol{u}, p_z) &\mapsto \begin{pmatrix} \frac{u_x - c_x}{f_x} p_z & \frac{u_y - c_y}{f_y} p_z & p_z \end{pmatrix}^\mathsf{T} \end{aligned} \tag{3.18}$$

3.3 Data Prefiltering

Before the captured data is further processed, a set of prefiltering steps is applied first. In addition to the aforementioned undistortion step, this primarily includes noise reduction which is usually performed by applying a bilateral filter [Tomasi and Manduchi, 1998] to the 2D depth image to obtain a smoothed version

$$z_{\text{filtered}}(\boldsymbol{u}) = \frac{\sum_{\boldsymbol{u}_i} K_{\text{range}}(\|z(\boldsymbol{u}_i) - z(\boldsymbol{u})\|_2)\, K_{\text{spatial}}(\|\boldsymbol{u}_i - \boldsymbol{u}\|_2)\, z(\boldsymbol{u})}{\sum_{\boldsymbol{u}_i} K_{\text{range}}(\|z(\boldsymbol{u}_i) - z(\boldsymbol{u})\|_2)\, K_{\text{spatial}}(\|\boldsymbol{u}_i - \boldsymbol{u}\|_2)} \tag{3.19}$$

where K_{spatial} denotes a filter kernel in the spatial pixel domain and K_{range} a kernel in the domain of the depth image values. A zero-mean Gaussian kernel $K(x) = \mathrm{e}^{-x^2/(2\sigma^2)}$, where the normalization constant can be omitted, is commonly applied for both kernels. Alternatively, Cao et al. [2018] used a temporal median filter where first a batch of consecutive depth images is registered (see Section 3.4) to determine pixel correspondences between the images and then the pixel-wise median over the batch is computed. In the context of unstructured point cloud data, such an initial denoising or consolidation step can be realized by, e.g., the LOP operator [Lipman et al., 2007] which performs L_1-based spatial denoising in 3D space and which we will study in more detail in Chapter 4.

Afterwards, a vertex map $\boldsymbol{p}(u_x, u_y)$ is commonly computed via back-projection with the pinhole camera model and subsequently used to derive further geometric information such as an estimate of the surface normals

$$\boldsymbol{n}(u_x, u_y) = \frac{(\boldsymbol{p}(u_x + 1, u_y) - \boldsymbol{p}(u_x - 1, u_y)) \times (\boldsymbol{p}(u_x, u_y + 1) - \boldsymbol{p}(u_x, u_y - 1))}{\left\|(\boldsymbol{p}(u_x + 1, u_y) - \boldsymbol{p}(u_x - 1, u_y)) \times (\boldsymbol{p}(u_x, u_y + 1) - \boldsymbol{p}(u_x, u_y - 1))\right\|_2} \tag{3.20}$$

in terms of finite differences, e.g. central differences as shown above.

3.4 Camera Tracking

Since the individual frames only capture certain parts of the scene from different angles, a crucial step in 3D reconstruction pipelines is the camera tracking stage where the data are brought into alignment by transforming each frame into a canonical world coordinate system. This transformation effectively determines the pose of the moving camera at a time step t and consists of a 3D rotation, which represents its orientation and can be defined in terms of a rotation matrix $\boldsymbol{R}^{(t)} \in \mathrm{SO}(3)$ with $\mathrm{SO}(3) = \{\boldsymbol{R} \in \mathbb{R}^{3\times3} \,|\, \boldsymbol{R}^\top \boldsymbol{R} = \mathbf{I}, \det(\boldsymbol{R}) = 1\}$, as well as a 3D translation $\boldsymbol{t}^{(t)} \in \mathbb{R}^3$, which denotes the position of the camera. A point $\boldsymbol{p}_{\text{camera}} \in \mathbb{R}^3$ from the local camera coordinate system can then be transformed into global world coordinates

$$\boldsymbol{p}_{\text{global}} = \boldsymbol{R}^{(t)}\, \boldsymbol{p}_{\text{camera}} + \boldsymbol{t}^{(t)} \tag{3.21}$$

by applying the rotation and translation. By writing the point in homogeneous coordinates $(\boldsymbol{p}_{\text{camera}}, 1)^\top \in \mathbb{R}^4$, the transformation can be defined in matrix form as

$$\boldsymbol{T}^{(t)} = \begin{pmatrix} \boldsymbol{R}^{(t)} & \boldsymbol{t}^{(t)} \\ \mathbf{0} & 1 \end{pmatrix} \in \mathbb{R}^{4\times4} \tag{3.22}$$

which allows for convenient inversion and concatenation of the transformations, e.g. to map between two local coordinate systems at time steps t_1 and t_2.

The estimation of these transformations from RGB-D frames as well as from other types of data such as pure monocular RGB images or additional inertial information has been an active field of research. Local tracking approaches optimize the relative pose $\Delta\boldsymbol{T}$ between two frames at time steps t and $t+1$ based on the energy formulation

$$E(\Delta\boldsymbol{T}) = E_{\text{geometric}}(\Delta\boldsymbol{T}) + \lambda_{\text{photometric}}\, E_{\text{photometric}}(\Delta\boldsymbol{T}) \tag{3.23}$$

consisting of a geometric term that is often accompanied with an additional photometric term and balanced by a weight $\lambda_{\text{photometric}} \in \mathbb{R}_{\geq 0}$. The geometric term only relies on the depth image and is usually defined in terms of point-to-plane distances in 3D space [Izadi et al., 2011; Newcombe et al., 2011; Nießner et al., 2013]

$$E_{\text{geometric}}(\Delta\boldsymbol{T}) = \sum_{\boldsymbol{u}} \langle \Delta\boldsymbol{T}\, \boldsymbol{p}^{(t+1)}(\boldsymbol{u}) - \boldsymbol{p}^{(t)}(\boldsymbol{u}_{\text{reprojected}}) \,|\, \boldsymbol{n}^{(t)}(\boldsymbol{u}_{\text{reprojected}}) \rangle^2 \tag{3.24}$$

where $\boldsymbol{u}_{\text{reprojected}} = \pi_{\boldsymbol{K}}([\boldsymbol{T}^{(t)}]^{-1}\, \boldsymbol{T}^{(t+1)}\, \boldsymbol{p}^{(t+1)}(\boldsymbol{u}))$ denotes the pixel $\boldsymbol{u}$ reprojected to the frame at time step t. On the other hand, the photometric term

$$E_{\text{photometric}}(\Delta\boldsymbol{T}) = \sum_{\boldsymbol{u}} \|I^{(t+1)}(\boldsymbol{u}) - I^{(t)}(\pi_{\boldsymbol{K}}(\Delta\boldsymbol{T}\, \boldsymbol{p}^{(t+1)}(\boldsymbol{u})))\|_2^2 \tag{3.25}$$

considers the intensity images $I^{(t)}$ and $I^{(t+1)}$, which are computed from the input RGB data, and penalizes differences between the reprojected and the observed intensity [Steinbrücker et al., 2011; Whelan et al., 2013]. This non-linear optimization problem is solved iteratively by linearizing the transformation $\Delta\boldsymbol{T}$ and obtaining the solution of the resulting linear least squares approximation. The final transformation at time step $t+1$ is then given by $\boldsymbol{T}^{(t+1)} = \boldsymbol{T}^{(t)} \cdot \Delta\boldsymbol{T}$. Whereas in frame-to-frame tracking two subsequent input frames are used to estimate the relative pose, in frame-to-model tracking the reference frame at time step t is generated from the current state of the reconstructed model which significantly reduces the amount of accumulated drift over time caused by slight estimation errors.

In order to ensure global consistency, various methods also considered the joint optimization of all transformations $E(\boldsymbol{T}^{(1)}, \boldsymbol{T}^{(2)}, \ldots, \boldsymbol{T}^{(t)})$ via loop closure detection and pose graph optimization techniques. For a more comprehensive overview, we refer to related surveys and state-of-the-art reports [Stotko, 2016b; Taketomi et al., 2017; Macario Barros et al., 2022]. In the scope of the methods developed in this thesis, we relied on the computationally faster frame-to-model methods.

3.5 Geometric Data Representations

After estimating the pose of the camera, the input frames can be merged into a 3D model. However, a naive composition into a single point cloud would lead to unpleasant, low-quality results since the individual frames do not only overlap and, hence, introduce significant redundancy, but are also corrupted by sensor noise and outliers. In order to obtain a smooth, compact, and high-quality estimate of the surface, we want to employ more sophisticated fusion approaches which require the notion of a suitable respective geometric data representation.

In the following, we briefly discuss common *explicit representations* such as points, surfels, and polygon meshes as well as *implicit representations* in terms of signed distance functions. A visual comparison of these representations is shown in Figure 3.2.

Points. One of the simplest surface representations is the point cloud model which consists of a set of unordered 3D points in space belonging to the surface:

$$\mathcal{S} := \{\boldsymbol{p}_i \in \mathbb{R}^3\} \tag{3.26}$$

A point cloud can effectively be considered as a discrete set of samples of the continuous surface and therefore captures a sparse subset of the whole geometry. As a consequence of this minimal representation, other properties including tangent planes, normals, or curvature information are not explicitly provided and must typically be estimated separately from the distribution of the points within a local neighborhood.

Surfels. In comparison to the point cloud model, *surface elements* (surfels) are primitives that are additionally equipped with further attributes of the surface which, in particular, includes first-order surface geometry information [Pfister et al., 2000]. To this end, surfel-based reconstruction approaches [Keller et al., 2013] typically maintain their data as an unordered list

$$\mathcal{S} := \{(\boldsymbol{p}_i, \boldsymbol{n}_i, r_i) \in \mathbb{R}^3 \times \mathbb{R}^3 \times \mathbb{R}_{\geq 0}\} \tag{3.27}$$

where each surfel is modeled as an oriented disk and specified by the center position $\boldsymbol{p}_i$, the orientation described by the normal vector $\boldsymbol{n}_i$, as well as the radius r_i of the disk. Furthermore, a color value or other auxiliary information that are relevant for the purpose of data fusion can be associated as well to a surfel.

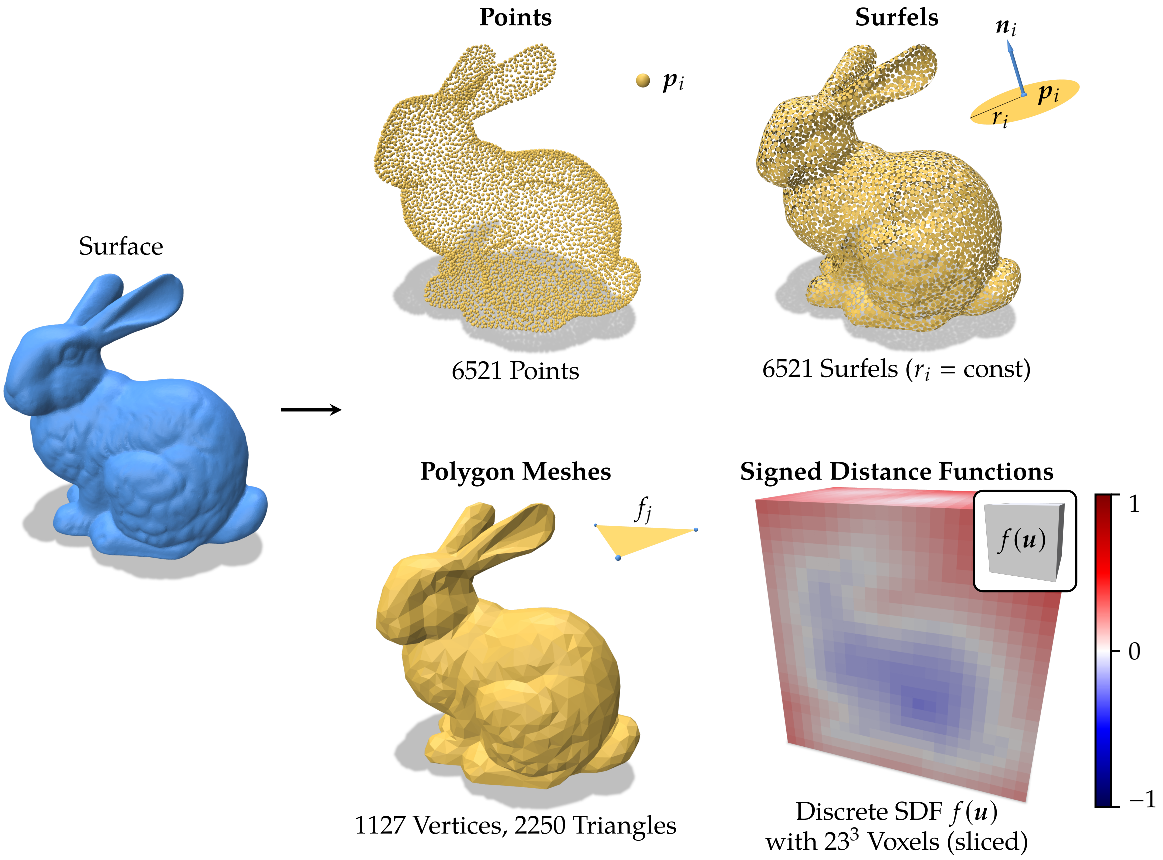

Points
p_i
Surfels
n_i
p_i
r_i
Surface
6521 Points
6521 Surfels (r_i = const)
Polygon Meshes
f_j
Signed Distance Functions
$f(u)$
1
0
−1
1127 Vertices, 2250 Triangles
Discrete SDF $f(u)$
with 23^3 Voxels (sliced)

Figure 3.2: Overview of common data model representations for surface reconstruction at the example of the *Bunny* model [Stanford Computer Graphics Laboratory, 1994]. Low model resolutions were chosen to visualize the individual structure of each explicit and implicit data representation.

Polygon Meshes. As points and surfels cover a small local part of the surface geometry in a flexible but also uncorrelated and unstructured way, the surface itself is not necessarily closed and may exhibit small holes in-between when it is rendered to an image. On the other hand, polygon meshes were utilized for several decades in graphics pipelines and offer a simple way to model closed surfaces as a graph

$$\mathcal{S} := (\mathcal{V}, \mathcal{F}) \quad \text{with} \quad \mathcal{V} := \{\boldsymbol{p}_i \in \mathbb{R}^3\}, \quad \mathcal{F} \subseteq \underbrace{\mathcal{V} \times \mathcal{V} \times \cdots \times \mathcal{V}}_{k \text{ times}} \tag{3.28}$$

where $\mathcal{V}$ denotes the set of vertices, i.e. the sampled points on the surface, and $\mathcal{F}$ the connectivity of the graph in terms of a set of faces. The most common choice are triangle meshes where $k = 3$ and the surface between the three vertices is represented by the respective triangle in a piecewise linear manner. In contrast to the other representations discussed in this section, polygon meshes are often only constructed in a post-processing step, e.g. via Marching Cubes [Lorensen and Cline, 1987] in the case of voxels (see Section 3.6.2). Some recent 3D reconstruction approaches also directly integrated an incremental meshing step into the data fusion stage to progressively update the reconstructed polygon mesh over time [Dong et al., 2018; Schöps et al., 2019b].

Signed Distance Functions. An alternative to the discussed explicit data models are implicit representations where, in particular, the *level set method* [Dervieux and Thomasset, 1979; Osher and Sethian, 1988] became very popular as an elegant framework to define and analyze surfaces. In its general form, the level set of a d-dimensional function $f : \mathbb{R}^d \to \mathbb{R}$ is defined as the set of all points $\boldsymbol{p} \in \mathbb{R}^d$ that are mapped to a constant value $c_{\text{level}} \in \mathbb{R}$ by f. For the purpose of 3D reconstruction where $d = 3$, we are interested in the zero level set

$$\mathcal{S} := \{\boldsymbol{p} \in \mathbb{R}^3 \,|\, f(\boldsymbol{p}) = 0\} \tag{3.29}$$

as a means to implicitly define the surface $\mathcal{S}$ in a continuous way. While in theory any function that fulfills the above requirement can be applied in this context, an intuitive choice for f is the *Signed Distance Function* (SDF) which returns the closest distance of a query point $\boldsymbol{p}$ to $\mathcal{S}$. Here, the sign of the distance indicates the location of the point relative to the surface. Points that are located within the object, i.e. lying behind the surface, have negative values $f(\boldsymbol{p}) < 0$ whereas they have positive values $f(\boldsymbol{p}) > 0$ if they lie outside the object and, thereby, in front of the surface. As we are only interested in the zero level set of the SDF, the other possible convention with flipped signs to determine the inner and outer space could be equally used instead. Nevertheless, we will use the former convention throughout this thesis since it has been adopted by the KinectFusion system [Izadi et al., 2011; Newcombe et al., 2011] and related techniques. Furthermore, we store the SDF discretely in *volume elements* (voxels). In fact, voxels can be considered as the generalization of 2D pixels to the three-dimensional domain where the 3D space is subdivided into axis-aligned cells that are associated with a scalar-valued or vector-valued property.

3.6 Volumetric Surface Reconstruction

After this brief overview of common geometric data representations, we will now discuss the surface reconstruction stage in more detail with a focus on volumetric methods based on signed distance functions which will serve as the basis for the techniques and systems developed in Chapters 5 to 7.

3.6.1 Projective Data Fusion

Given a set of n depth images which capture the surface from various angles, Curless and Levoy [1996] formulated the objective of fusing the given overlapping depth information into a single consistent signed distance function that best approximates the observations as an optimization problem. To this end, the respective energy function has been defined as a weighted linear least squares problem

$$E(\mathcal{S}) = \sum_{t=1}^{n} \int_{\Omega^{(t)}} w_{\text{sight}}^{(t)}(\boldsymbol{x}, \mathcal{S})\, d_{\text{sight}}^{(t)}(\boldsymbol{x}, \mathcal{S})^2 \,\mathrm{d}\boldsymbol{x} \tag{3.30}$$

where for each depth image captured at time step t the signed distances $d_{\text{sight}}^{(t)}$ of the measured 3D points to the surface along the line of sight of the sensor are minimized. Here, the integration domain $\Omega^{(t)}$ is specified with respect to the line of sight and depends on the time step t as well as on the corresponding sensor pose estimated in the tracking stage (see Section 3.4). By reparametrizing the energy to a canonical domain, the global SDF that minimizes the above energy can be computed as the weighted average

$$D^{*}(\boldsymbol{p}) = \frac{\sum_{t=1}^{n} w^{(t)}(\boldsymbol{p})\, d^{(t)}(\boldsymbol{p})}{\sum_{i=1}^{n} w^{(t)}(\boldsymbol{p})} \tag{3.31}$$

of signed distances $d^{(t)}$ for every point $\boldsymbol{p} \in \mathbb{R}^3$. This solution can be further reformulated to an incremental update step

$$D^{(t+1)}(\boldsymbol{p}) = \frac{W^{(t)}(\boldsymbol{p})\, D^{(t)}(\boldsymbol{p}) + w^{(t)}(\boldsymbol{p})\, d^{(t)}(\boldsymbol{p})}{W^{(t)}(\boldsymbol{p}) + w^{(t)}(\boldsymbol{p})} \tag{3.32}$$

$$W^{(t+1)}(\boldsymbol{p}) = W^{(t)}(\boldsymbol{p}) + w^{(t)}(\boldsymbol{p}) \tag{3.33}$$

which avoids the cost of storing all observations and allows to process the data in an online fashion. In practice, the signed distances $d^{(t)}$ are computed in a projective way as the differences between the depth of the points $\boldsymbol{p}$ and the corresponding depth measurements at the projected image positions of $\boldsymbol{p}$. Furthermore, the values are truncated and specified relative to a small band $c_{\text{truncation}} \in \mathbb{R}_{>0}$ since only the close proximity around the surface is required to accurately represent and store it. Therefore, a *Truncated Signed Distance Function* (TSDF) is effectively computed. Further information such as colors provided by the RGB camera of the sensor or albedo information (see Section 3.7.2) can be fused as well using the same incremental update step.

3.6.2 Surface Extraction via Marching Cubes

Although many operations for signed distance functions including data fusion can be performed very efficiently and elegantly, directly rendering an image of the implicitly defined surface is, however, a more complex operation and requires to traverse the volume along a set of cast rays and to repeatably sample the SDF. In contrast to this, the visualization of explicit representations such as polygon meshes usually only requires a rasterization step which is a standard operation for rendering and implemented in a highly optimized manner on the GPU. To this end, Marching Cubes [Lorensen and Cline, 1987] has been developed as an approach to extract the isosurface at a given value $f(\boldsymbol{p}) = c_{\text{iso}}$ in terms of a triangle mesh. This algorithm will be a key component of our proposed telepresence system which

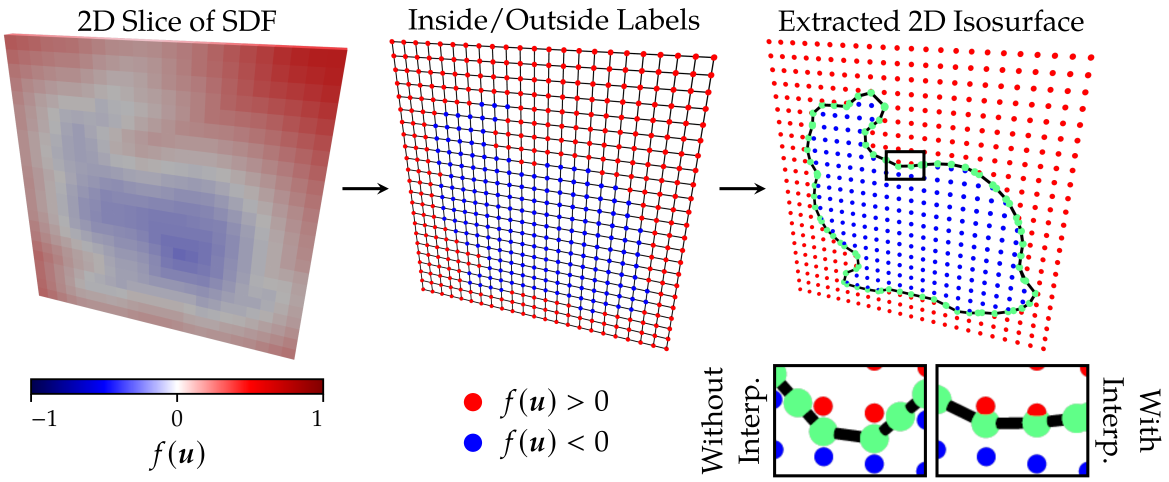


Figure 3.3: Visualization of the Marching Cubes algorithm [Lorensen and Cline, 1987] at the example of the *Bunny* model [Stanford Computer Graphics Laboratory, 1994] for a single 2D slice of the voxel grid, inspired by Anderson [2004].

we will discuss in Chapters 5 and 6. A visual overview of the algorithm for a 2D slice of a volumetrically stored SDF is shown in Figure 3.3.

Starting from an arbitrary voxel with discrete coordinates $\boldsymbol{u} \in \mathbb{Z}^3$, the stored SDF values $d_0, \dots, d_7$ of the voxel and its seven neighboring voxels at the positions

$$\mathcal{N}_{\mathrm{MC}}(\boldsymbol{u}) = \{(u_x + \Delta u_x, u_y + \Delta u_y, u_z + \Delta u_z)^\top \in \mathbb{Z}^3 \mid \Delta \boldsymbol{u} \in \{0,1\}^3\} \tag{3.34}$$

are fetched and compared against the specified isovalue $c_{\mathrm{iso}} \in \mathbb{R}$ to determine whether the isovalue is crossed in-between the discrete voxels. This results in a binary code of eight bits that can be written as an index $i_{\mathrm{MC}} \in [0, 255] \subset \mathbb{N}_0$ encoding the detected configuration of the local surface structure. Based on this configuration, vertices are placed at the edges between the voxels crossing the isovalue and connected to a set of triangles. In practice, this is efficiently implemented in terms of a precomputed lookup table which can be indexed by i_{MC} and returns the involved edges as well as the triangulation pattern for any possible configuration. Since the concrete values of the SDF are available, the positions along the edges can be linearly interpolated to obtain a smoother mesh that more closely approximates the isosurface.

3.6.3 Spatial Voxel Block Hashing

Although the KinectFusion system [Izadi et al., 2011; Newcombe et al., 2011] allows to run the aforementioned stages of the 3D reconstruction pipeline in real time, one of its major limitations was the runtime and storage complexity which restricts the system to room-scale scenes. Due to its reliance on managing a dense voxel grid, doubling the size of the scene or the resolution of the grid would result in eight times higher memory requirements and, hence, a cubic scaling. In order to address this problem, Nießner et al. [2013] proposed an extension where only a small band around the surface was actively stored which reduced the requirements to almost quadratic complexity and was close to the theoretically optimal asymptotic scaling. In particular, a two-level voxel grid hierarchy is employed where the finer level stores the actual voxel data similar to KinectFusion and the coarser level provides a lightweight mechanism to handle sets of $8^3 = 512$ voxels by composing them into blocks. These blocks are managed by a GPU hash table and can be retrieved by evaluating the spatial hash function [Teschner et al., 2003]

$$\begin{aligned} h_{\text{block}} \colon \mathbb{Z}^3 &\to \mathbb{N}_0 \\ \boldsymbol{u} &\mapsto (u_x \cdot p_1 \oplus u_y \cdot p_2 \oplus u_z \cdot p_3) \mod n \end{aligned} \tag{3.35}$$

where the coordinates of the discrete 3D block position $\boldsymbol{u}$ are first multiplied by the large prime numbers $p_1 = 73856093$, $p_2 = 19349669$, $p_3 = 83492791$, then merged into a single value using the XOR operator $\oplus$, and finally mapped to one of the n hash table buckets. Applying this sparse volumetric grid representation to the reconstruction pipeline requires several further changes and additional steps which will be described in the following.

Block Allocation. Since only a tight band around the reconstructed surface should be explicitly stored and maintained, the set of voxel blocks, that lie within this region and should be subsequently updated, has to be computed first. To this end, the pixel-wise range of the band is determined from the depth image values z as the interval $[z - c_{\text{truncation}}, z + c_{\text{truncation}}]$ and the corresponding line segments are constructed via back-projection of the interval bounds (see Section 3.2.2). Afterwards, the Digital Differential Analyzer (DDA) algorithm [Amanatides and Woo, 1987] is applied to find all blocks that intersect with these segments and each block is allocated and inserted into the GPU hash table.

Block Garbage Collection. Some depth samples are potentially unreliable due to noise and may unnecessarily increase the size of the truncation band. Such voxel blocks are typically located at the border of the band and will receive significantly less updates than blocks that are close to the surface. Therefore, the maximum weight as well as the minimum absolute SDF value within a block are computed and compared against some thresholds. Blocks that are detected in this way are removed from the hash table and deallocated.

Block Streaming. In addition to the more compact storage of the surface, the unordered structure of the sparse voxel grid in terms of spatially-hashed blocks enables further possibilities to improve the scalability of the overall system by streaming currently inactive blocks from the smaller GPU memory to the larger CPU memory. This is determined by testing whether the considered block is still visible to the moving (virtual) camera. Once a streamed-out voxel block becomes visible again and its voxel data will be updated with new sensor information, the block is streamed back into GPU memory. Nießner et al. [2013] managed the blocks on the CPU in larger groups of chunks using a separate linked list, which effectively corresponds to a three-level hierarchy, whereas Kähler et al. [2015] reused the GPU hash table and instead tracked the state of the block explicitly by a flag.

3.7 Image-based Appearance Reconstruction

Although the surface geometry provides clues about the shape of the reconstructed objects and scenes, it lacks information about the perceived appearance which is another crucial ingredient to reconstruct realistic 3D models. Before we discuss the preliminaries of image-based appearance reconstruction in terms of intrinsic image decomposition, which will be the focus of our work discussed in Chapter 7, we first introduce the fundamental concepts of light transport.

3.7.1 Light Transport

Understanding how the final color values in an RGB image are determined from the captured environment requires knowledge about the interaction of light with the scene and, thereby, about light transport through the scene. A general formulation of this physical process has been introduced by Kajiya [1986] in terms of the *Rendering Equation*. Considering only non-translucent, opaque objects in the scene, the radiance $\boldsymbol{L}_o$, that is emitted or reflected at a surface point $\boldsymbol{p} \in \mathcal{S}$ into the direction $\boldsymbol{\omega}_o \in \mathcal{H}(\boldsymbol{n})$ of the hemisphere $\mathcal{H}(\boldsymbol{n})$, can be formulated as

$$\boldsymbol{L}_o(\boldsymbol{p}, \boldsymbol{\omega}_o) = \boldsymbol{L}_e(\boldsymbol{p}, \boldsymbol{\omega}_o) + \int_{\mathcal{H}(\boldsymbol{n})} \boldsymbol{f}_{\mathrm{BRDF}}(\boldsymbol{p}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o)\, \boldsymbol{L}_i(\boldsymbol{p}, \boldsymbol{\omega}_i)\, \langle \boldsymbol{n} | \boldsymbol{\omega}_i \rangle \, \mathrm{d}\boldsymbol{\omega}_i \tag{3.36}$$

and consists of an emission term and a reflection term. Here, the former term specifies the directly emitted radiance $\boldsymbol{L}_e$ from the surface, which is non-zero for light sources, whereas the latter term denotes the total amount of reflected light and, hence, captures all contributions of reflected light over the hemisphere $\mathcal{H}(\boldsymbol{n})$. Each of these individual contributions is determined by the total amount of incoming radiance $\boldsymbol{L}_i$ received from the direction $\boldsymbol{\omega}_i \in \mathcal{H}(\boldsymbol{n})$, the *Bidirectional Reflection Distribution Function* (BRDF) $\boldsymbol{f}_{\mathrm{BRDF}}$, as well as an additional attenuation term based on the angle of the incident light to the surface. The BRDF $\boldsymbol{f}_{\mathrm{BRDF}}(\boldsymbol{p}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o)$ is a property of the material and describes the fraction of light that is reflected from a direction $\boldsymbol{\omega}_i$ to a direction $\boldsymbol{\omega}_o$.

Since radiance remains constant during light transport through vacuum, the incoming radiance $\boldsymbol{L}_i(\boldsymbol{p}, \boldsymbol{\omega}_i)$ at the point $\boldsymbol{p}$ is equal to the outgoing radiance $\boldsymbol{L}_o(\boldsymbol{p}', -\boldsymbol{\omega}_i)$ at the point $\boldsymbol{p}'$ which denotes the closest intersection point of the ray $\boldsymbol{l}(\tau) = \boldsymbol{p} + \tau\, \boldsymbol{\omega}_i$ with the scene. This assumption can also be applied in many real-world scenarios that do not include fog or other complex participating media. As a result, the rendering equation can be expressed as a recursive integral equation which, in the general case, has no analytic closed-form solution.

3.7.2 Intrinsic Image Decomposition

Whereas the rendering equation formally defines the interaction of known scene geometry, material properties and surrounding illumination which is primarily used to generate photo-realistic images, inverse rendering describes the process of recovering these components from the final images. A simple, yet effective formulation of this process in image space is *intrinsic* *image decomposition* introduced by Barrow and Tenenbaum [1978] which separates a captured image into an element-wise product of reflectance and shading images and can be derived from the rendering equation under the consideration of additional assumptions [Bonneel et al., 2017; Garces et al., 2022]. To this end, the reflection behavior of a material and, thereby, its BRDF can be analyzed by splitting it into a diffuse and a specular term

$$f_{\mathrm{BRDF}}(\boldsymbol{p}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o) = \boldsymbol{\kappa}_{\mathrm{d}}(\boldsymbol{p})\, f_{\mathrm{d}}(\boldsymbol{p}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o) + \boldsymbol{\kappa}_{\mathrm{s}}(\boldsymbol{p})\, f_{\mathrm{s}}(\boldsymbol{p}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o) \tag{3.37}$$

where $\boldsymbol{\kappa}_{\mathrm{d}} \in [0,1]^3$ and $\boldsymbol{\kappa}_{\mathrm{s}} \in [0,1]^3$ denote the respective albedo components of the material and f_{d} and f_{s} the diffuse and specular lobes of the BRDF [Guarnera et al., 2016]. In the context of intrinsic image decomposition, the materials are assumed to be Lambertian and only reflect light in a perfectly diffuse manner without further specular terms. This results in the Lambert BRDF $f_{\mathrm{BRDF}}(\boldsymbol{p}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o) = \boldsymbol{\kappa}_{\mathrm{d}}(\boldsymbol{p})/\pi$ [Lambert, 1760] which distributes radiance uniformly into all directions and does not depend on the incoming or outgoing directions $\boldsymbol{\omega}_i$ and $\boldsymbol{\omega}_o$. If we further assume that the captured scene is not additionally emitting light, i.e. no light sources are directly visible in the final image, the rendering equation reduces to

$$\boldsymbol{L}_o(\boldsymbol{p}, \boldsymbol{\omega}_o) = \boldsymbol{\kappa}_{\mathrm{d}}(\boldsymbol{p}) \cdot \int_{\mathcal{H}(\boldsymbol{n})} \frac{1}{\pi}\, \boldsymbol{L}_i(\boldsymbol{p}, \boldsymbol{\omega}_i)\, \langle \boldsymbol{n} | \boldsymbol{\omega}_i \rangle \, \mathrm{d}\boldsymbol{\omega}_i \tag{3.38}$$

and reveals the separation of the diffuse albedo from the remaining shading terms for each surface point $\boldsymbol{p}$. In image space, this relation denotes the classic intrinsic image decomposition formula

$$\boldsymbol{L}(\boldsymbol{u}) = \boldsymbol{\kappa}_{\mathrm{d}}(\boldsymbol{u})\, \boldsymbol{s}(\boldsymbol{u}) \tag{3.39}$$

where the diffuse albedo $\boldsymbol{\kappa}_{\mathrm{d}}$ is typically referred to as the reflectance term and $\boldsymbol{s}$, representing the integral term, is called the shading term. Here, it is important to note that during the image formation process the linear radiance values are converted to gamma space and finally mapped to pixel values via quantization. Therefore, the decomposition is formally defined in gamma space which more closely resembles the human vision system and allows to interpret differences in color on a perceptional basis.

Computing the actual decomposition remains a challenging and ill-posed problem due to the inherent pixel-wise scaling ambiguity which implies that there exists an infinite number of solutions $(\boldsymbol{\kappa}_{\mathrm{d}} \cdot \boldsymbol{c}_{\mathrm{random}}, \boldsymbol{s}/\boldsymbol{c}_{\mathrm{random}})$ that perfectly explain the final image $\boldsymbol{L}$ after re-composition. Several priors have been employed to constrain the problem and to resolve this ambiguity where popular choices are based on the *Retinex Theory* [Land and McCann, 1971] or the assumption of white light to restrict the shading term to a one-dimensional scalar value.

Part II

Publications

Chapter 4

Incomplete Gamma Kernels: Generalizing Locally Optimal Projection Operators

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:

> Patrick Stotko, Michael Weinmann, and Reinhard Klein.
> "Incomplete Gamma Kernels: Generalizing Locally Optimal Projection Operators."
> *arXiv:2205.01087 (under review), submitted to IEEE Transactions of Pattern Analysis and Machine Intelligence (TPAMI)*, 2022.
> DOI: `10.48550/arXiv.2205.01087`

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

Author Contributions of the Publication In this work, I developed the theoretical derivation of the kernel family, its properties, as well as the applications. Furthermore, I performed the experiments and evaluations of the proposed extensions.

4.1 Abstract

We present incomplete gamma kernels, a generalization of Locally Optimal Projection (LOP) operators. In particular, we reveal the relation of the classical localized L_1 estimator, used in the LOP operator for surface reconstruction from noisy point clouds, to the common Mean Shift framework via a novel kernel. Furthermore, we generalize this result to a whole family of kernels that are built upon the incomplete gamma function and each represents a localized L_p estimator. By deriving various properties of the kernel family concerning distributional, Mean Shift induced, and other aspects such as strict positive definiteness, we obtain a deeper

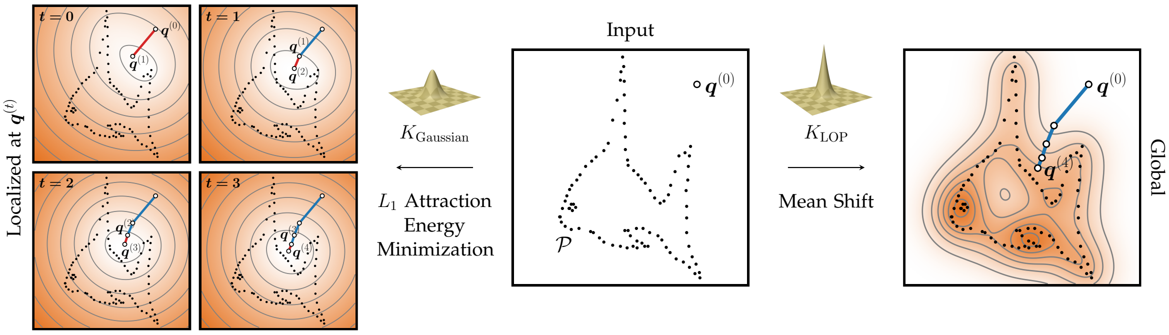


Figure 4.1: Relation between LOP and Mean Shift at the example of the *2D Fish* model. Minimizing the *localized* L_1 attraction energy with the Gaussian kernel K_{Gaussian} (*left*) results in the same trajectory $\boldsymbol{q}^{(t)}$ as applying Mean Shift on a *global* kernel density estimate with the kernel K_{LOP} (*right*).

understanding of the operator's projection behavior. From these theoretical insights, we illustrate several applications ranging from an improved Weighted LOP (WLOP) density weighting scheme and a more accurate Continuous LOP (CLOP) kernel approximation to the definition of a novel set of robust loss functions. These incomplete gamma losses include the Gaussian and LOP loss as special cases and can be applied for reconstruction tasks such as normal filtering. We demonstrate the effects of each application in a range of quantitative and qualitative experiments that highlight the benefits induced by our modifications.

4.2 Introduction

Digital 3D scene models have become a crucial prerequisite for numerous applications in entertainment, advertisement, design, architecture, autonomous systems, and cultural heritage. In this context, the accurate digitization of real-world objects and scenes is of great relevance and offers new opportunities regarding a variety of tasks including AR/VR-based inspection and collecting realistic training data for tasks in robotics, autonomous driving, aerial or satellite surveys. Aside from professional scanning campaigns with expensive laser scanning equipment, there has also been an increasing trend towards more practical scene capture with consumer-grade hardware such as passive purely image-based scene scanning using Structure-from-Motion and Multi-view Stereo approaches, or with respective cheaper active time-of-flight depth sensors that have meanwhile even been integrated into numerous mobile devices. However, the use of passive scene scanning or active scanning based on cheap hardware with low sensor quality and low sensor resolution induces noise in both the capture process and the 3D reconstruction procedure, and thereby results in noisy point clouds and a low number of points that might not preserve finer geometric details in the reconstruction respectively, which, in turn, may lead to registration artifacts. Furthermore, the limited accessibility of capture conditions as well as occlusions induce holes in the reconstructed models. These challenges result in an increasing interest in robust surface reconstruction techniques capable of handling noise, outliers, registration artifacts and missing data.

Among others, the Locally Optimal Projection (LOP) operator [Lipman et al., 2007] has gained a lot of attention in recent years due to its benefit of not relying on a well-defined surface parametrization or a piecewise planar approximation and, meanwhile, there has been a whole series of further extensions of this approach [Huang et al., 2009; Huang et al., 2013; Liao et al., 2013; Preiner et al., 2014]. Furthermore, many learning-based approaches also aim at projecting the noisy data onto a (latent) denoised manifold [Zhang et al., 2020; Xu et al., 2022a]. Therefore, investigations towards the unification of traditional approaches with their respective regularization techniques might be of great relevance for future learning-based approaches as well. Even further, traditional techniques such as Mean Shift clustering [Fukunaga and Hostetler, 1975; Cheng, 1995; Comaniciu and Meer, 2002] become more and more relevant in modern deep learning methods. Besides their application to structure the latent space representation of the data within encoder-decoder approaches [Madaan et al., 2019], there is even a direct relation between the Mean Shift approach and denoising autoencoders [Bigdeli and Zwicker, 2018]. In particular, as the output of an optimal denoising autoencoder corresponds to the local mean of the true data density [Alain and Bengio, 2014], the autoencoder loss can be interpreted as a Mean Shift vector [Bigdeli and Zwicker, 2018]. However, to the best of our knowledge, this observation has not yet been explored in the context of surface reconstruction and denoising. Hence, relating traditional concepts to modern deep learning methods might not only lead to a more explainable behavior of the latter but also allow increasing the resulting performance. In turn, this relies on the better understanding of the relationship between previous (traditional) techniques.

In this paper, we investigate the theoretical relationship of projection-based surface reconstruction approaches with their respective properties and show that these are unified within the common Mean Shift framework. In particular, the key contributions of our work are:

- We reveal the relation of the classical localized L_1 estimator used in LOP to the Mean Shift framework via a novel kernel K_{LOP} and introduce the family of *incomplete gamma kernels* K_Γ as a generalization of this result where each kernel represents a localized L_p estimator (see Section 4.4).

- We derive various properties of the kernel family concerning distributional, Mean Shift induced, and other aspects such as strict positive definiteness to obtain a deeper understanding of the operator's projection behavior (see Section 4.5).

- We demonstrate that leveraging the derived theoretical insights enables several applications including an improved Weighted LOP (WLOP) density weighting scheme, a more accurate Continuous LOP (CLOP) kernel approximation as well as the derivation of *incomplete gamma losses*, a set of novel robust loss functions (see Section 4.6).

In our evaluation, we demonstrate the benefits induced by our modifications in a range of quantitative and qualitative experiments. Furthermore, the theoretical insights of our investigations with their proven effect may be of great relevance also for future learning-based approaches.

4.3 Related Work

In the following, we provide a review of geometric and learning-based denoising approaches. Furthermore, we also review seminal work regarding the theory and application of the Mean Shift framework due to its relationship to LOP approaches that we will demonstrate later.

4.3.1 Geometric Denoising Approaches

Following early approaches such as the local fitting of tangent planes [Hoppe et al., 1992] or using radial basis functions [Carr et al., 2001], respective developments particularly focused on projection-based methods, sparsity-based methods and non-local methods.

Projection-based Methods. These approaches rely on the assumption of an underlying smooth surface and the projection of noisy data points onto the estimated local surface. For this purpose, respective approaches apply moving least squares (MLS) based methods [Alexa et al., 2003; Amenta and Kil, 2004; Fleishman et al., 2005; Öztireli et al., 2009], robust principal component analysis (RPCA) [Narváez and Narváez, 2006] and moving robust principal component analysis (MRPCA) [Mattei and Castrodad, 2017], or locally optimal projection based operators where the LOP operator [Lipman et al., 2007] has been extended in terms of Weighted LOP (WLOP) [Huang et al., 2009], Feature LOP (FLOP) [Liao et al., 2013], Continuous LOP (CLOP) [Preiner et al., 2014], Edge-Aware Resampling (EAR) [Huang et al., 2013] and a Gaussian mixture model inspired projection operator [Lu et al., 2017]. The latter has been demonstrated to be capable of resampling point clouds while preserving features due to the additional guidance of filtered normals.

Sparsity-based Methods. This class of approaches relies on the assumption that objects can be represented in terms of piecewise smooth surfaces with sparse features. Respective denoising techniques include L_0-norm [Sun et al., 2015; Cheng et al., 2019a] and L_1-norm minimization [Avron et al., 2010; Mattei and Castrodad, 2017; Leal et al., 2020], sparse dictionary learning [Digne et al., 2017] as well as patch-based or feature-based graph Laplacian regularization [Zeng et al., 2019; Dinesh et al., 2020; Hu et al., 2020], graph-based point cloud denoising based on jointly leveraging geometry and color information [Irfan and Magli, 2021], guided filtering based on normal information followed by a L_1-medial skeleton extraction to get the sharp structure of the surface [Zheng et al., 2017] as well as leveraging gravitational feature functions [Shi et al., 2022]. In the context of denoising dynamic point clouds, Hu et al. [2021] explored the temporal coherence of spatio-temporal graphs with respect to the underlying surface, where a respective manifold-to-manifold distance has been introduced. Furthermore, data-driven exemplar priors have been used for surface reconstruction [Remil et al., 2017], where the sparsity of local shapes from a collection of 3D objects has been explored.

Non-local Methods. In contrast to the previous classes, these approaches rely on the assumption that geometric statistics are (approximately) shared by certain surface patches of a 3D model, i.e. local surface denoising is conducted based on collected neighborhoods with similar geometry [Rosman et al., 2013; Chen et al., 2019; Lu et al., 2020; Zhu et al., 2022]. However, the definition of a suitable metric as well as the regular representation of local surface structures remain challenging. Furthermore, density-based point cloud denoising has been approached by first applying particle-swarm based optimization for kernel density estimation followed by a Mean Shift clustering-based outlier removal and a final bilateral mesh filtering [Zaman et al., 2017].

4.3.2 Learning-based Denoising Approaches

Recent works more and more leverage deep learning for surface reconstruction from point clouds as well as point cloud denoising. Examples include approaches for point cloud consolidation and resampling such as PointNet [Qi et al., 2017a], PointNet++ [Qi et al., 2017b], patch-based progressive point cloud upsampling [Yifan et al., 2019b] as well as the unification of the considerations of densifying, denoising and completing point clouds [Choe et al., 2022]. Other approaches followed the principles of initially projecting the points onto coarse-level local reference planes and applying a subsequent refinement [Duan et al., 2019] or the initial removal of outliers before conducting the denoising [Rakotosaona et al., 2020]. Further approaches include edge-aware point cloud consolidation [Yu et al., 2018], adversarial defense [Zhou et al., 2019], graph-convolutional methods [Pistilli et al., 2020], unsupervised approaches such as Total Denoising [Hermosilla et al., 2019], gradient field based denoising [Chen et al., 2021a; Luo and Hu, 2021; Zhao et al., 2022], differentiable approaches [Roveri et al., 2018; Yifan et al., 2019a; Luo and Hu, 2020] as well as manifold learning based on encoder-decoder architectures [Zhang et al., 2020; Xu et al., 2022a]. Non-local self-similarities have also been considered to define neural self-priors that capture geometric repetitions [Hanocka et al., 2020], capture semantically related non-local features [Huang et al., 2020a], or apply self-correction by allowing the model to capture structural and contextual information from initially disorganized parts [Chen et al., 2021b]. Furthermore, normalizing flows have been applied to the learn the distribution of noisy points and disentangle noise from the latent space [Mao et al., 2022]. In addition, the feature-aware recurrent point cloud denoising network (RePCD-Net) [Chen et al., 2022a] combines a recurrent network architecture for noise removal with multi-scale feature aggregation and propagation and a feature-aware Chamfer distance loss.

4.3.3 Mean Shift Approaches

The Mean Shift approach [Fukunaga and Hostetler, 1975] is a well-studied local mode-seeking method with diverse applications including data clustering [Cheng, 1995; Comaniciu and Meer, 2002; Grillenzoni, 2016; Beck et al., 2019], image filtering [Comaniciu and Meer, 2002], segmentation [Comaniciu and Meer, 2002; Jang and Jiang, 2021], denoising [Bigdeli et al., 2017; Bigdeli and Zwicker, 2018], and object tracking [Jang and Jiang, 2021]. Tremendous effort has been spent to study its convergence behavior [Cheng, 1995; Comaniciu and Meer, 2002; Li et al., 2007; Chen, 2015; Ghassabeh, 2015; Huang et al., 2018] which culminated in a rigorous set of properties proven by Yamasaki and Tanaka [Yamasaki and Tanaka, 2019]. Recently, Mean Shift clustering has also been applied in the latent space of neural encoder-decoder approaches to achieve a better structured data representation [Madaan et al., 2019]. Furthermore, the connection between the Mean Shift approach and denoising autoencoders [Vincent et al., 2008] has been revealed by Bigdeli and Zwicker [2018], who leveraged the observation that the output of an optimal denoising autoencoder (DAE) is a local mean of the true data density [Alain and Bengio, 2014] to show that that the autoencoder loss is a Mean Shift vector and to use the respective magnitude to define a prior for image restoration.

4.4 Background

Before deriving our proposed kernel family as a generalization of LOP in the context of Mean Shift, we first provide a brief introduction into the concepts of both approaches.

4.4.1 Mean Shift

The basic objective of Mean Shift [Fukunaga and Hostetler, 1975] is to find the modes of a density function f which has been observed by a (sparse) set of points $\mathcal{P} = \{\boldsymbol{p}_i \in \mathbb{R}^d\}$ and is modeled by a kernel density estimate function:

$$\hat{f}_{\mathcal{P},K}(\boldsymbol{q}) = \frac{1}{|\mathcal{P}|\, h^d} \sum_i K(\tfrac{\boldsymbol{p}_i - \boldsymbol{q}}{h}) \tag{4.1}$$

Here, h denotes the kernel window size and K a kernel that is non-negative ($K(\boldsymbol{x}) \geq 0$), normalized ($\int_{\mathbb{R}^d} K(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = 1$), and radially symmetric ($K(\boldsymbol{x}) = c_K\, k(\|\boldsymbol{x}\|^2)$). The function k defined in the symmetry constraint along with the normalization constant c_K is called the kernel profile of K and plays an important role in the analysis of Mean Shift [Yamasaki and Tanaka, 2019]. Furthermore, the gradient of the kernel density estimate

$$\nabla \hat{f}_{\mathcal{P},K}(\boldsymbol{q}) = \frac{2}{|\mathcal{P}|\, h^{d+2}} \frac{c_K}{c_G} \sum_i G(\tfrac{\boldsymbol{p}_i - \boldsymbol{q}}{h})\,(\boldsymbol{p}_i - \boldsymbol{q}) \tag{4.2}$$

can be derived using the kernel $G(\boldsymbol{x}) = c_G\, g(\|\boldsymbol{x}\|^2)$ and its corresponding profile $g(x) = -\frac{\mathrm{d}}{\mathrm{d}x} k(x)$. Based on these two functions, Mean Shift finds the modes by iteratively applying the Mean Shift vector

$$\boldsymbol{m}_{\mathcal{P},G}(\boldsymbol{q}) = \frac{h^2}{2} \frac{c_G}{c_K} \frac{\nabla \hat{f}_{\mathcal{P},K}(\boldsymbol{q})}{\hat{f}_{\mathcal{P},G}(\boldsymbol{q})} = \frac{\sum_i G(\frac{\boldsymbol{p}_i - \boldsymbol{q}}{h})\,(\boldsymbol{p}_i - \boldsymbol{q})}{\sum_i G(\frac{\boldsymbol{p}_i - \boldsymbol{q}}{h})} \tag{4.3}$$

which describes the gradient vector normalized with respect to the kernel G and is the main component in the algorithm. In particular, it directly defines the update step of the corresponding fixed-point iteration $\boldsymbol{q}^{(t+1)} = \boldsymbol{q}^{(t)} + \boldsymbol{m}_{\mathcal{P},G}(\boldsymbol{q}^{(t)})$ which performs gradient ascent on the kernel density estimate function $\hat{f}_{\mathcal{P},K}$.

4.4.2 Locally Optimal Projection

The central tendency of a set of data points, often measured in terms of the geometric mean, is a crucial and desirable property used in many applications, but its definition and computation has been a challenging research question for decades. Although the geometric mean can be easily evaluated, it is highly sensitive to outliers. In contrast, the L_1 median — also called geometric median — is a more robust quantity and can be computed by the iterative Weiszfeld algorithm [Weiszfeld, 1937]. In addition to many other contexts, it has been applied to 3D surface reconstruction to define a robust projection operator [Lipman et al., 2007]. Given a set of noisy target points $\mathcal{P} = \{\boldsymbol{p}_i \in \mathbb{R}^d\}$ sampled from a surface $\mathcal{S}$ where $d = 3$ is usually considered, the task consists in projecting an additional set of projection points $\boldsymbol{Q} = \{\boldsymbol{q}_j \in \mathbb{R}^d\}$ onto $\mathcal{S}$ based on the observations $\mathcal{P}$. This can be expressed in terms of an energy formulation

$$E(\boldsymbol{Q}) = \sum_j E_{\text{LOP}}(\boldsymbol{q}_j) + E_{\text{rep}}(\boldsymbol{q}_j) \tag{4.4}$$

based on an attraction and a repulsion term

$$E_{\text{LOP}}(\boldsymbol{q}_j) = \sum_i \theta(\|\boldsymbol{p}_i - \boldsymbol{q}_j^{(t)}\|)\, \|\boldsymbol{p}_i - \boldsymbol{q}_j\| \tag{4.5}$$

$$E_{\text{rep}}(\boldsymbol{q}_j) = \lambda_j \sum_{i, i \neq j} \theta(\|\boldsymbol{q}_i^{(t)} - \boldsymbol{q}_j^{(t)}\|)\, \eta(\|\boldsymbol{q}_i^{(t)} - \boldsymbol{q}_j\|) \tag{4.6}$$

where $\theta(x) = \mathrm{e}^{-x^2/(h/4)^2}$ denotes a compact localization kernel and η a decreasing regularization function penalizing small distances between projection points. Common choices of η include the originally proposed function $\eta_{\text{LOP}}(x) = 1/(3x^3)$ [Lipman et al., 2007] as well as the less rapidly decreasing function $\eta_{\text{WLOP}}(x) = -x$ [Huang et al., 2009]. Both energy terms are balanced by weights λ_j which are chosen such that they only depend on a single, global parameter $\mu \in [0, 1/2)$. Based on the Weiszfeld algorithm, the solution to this optimization problem can be obtained by the fixed-point iteration

$$\begin{aligned} \boldsymbol{q}_j^{(t+1)} = {} & \frac{\sum_i \alpha(\|\boldsymbol{p}_i - \boldsymbol{q}_j^{(t)}\|)\, \boldsymbol{p}_i}{\sum_i \alpha(\|\boldsymbol{p}_i - \boldsymbol{q}_j^{(t)}\|)} \\ & + \mu \, \frac{\sum_{i, i \neq j} \beta(\|\boldsymbol{q}_i^{(t)} - \boldsymbol{q}_j^{(t)}\|)\, (\boldsymbol{q}_j^{(t)} - \boldsymbol{q}_i^{(t)})}{\sum_{i, i \neq j} \beta(\|\boldsymbol{q}_i^{(t)} - \boldsymbol{q}_j^{(t)}\|)} \end{aligned} \tag{4.7}$$

with kernels $\alpha(x) = \theta(x)/x$ and $\beta(x) = \theta(x)/x \left| \frac{\mathrm{d}}{\mathrm{d}x} \eta(x) \right|$.

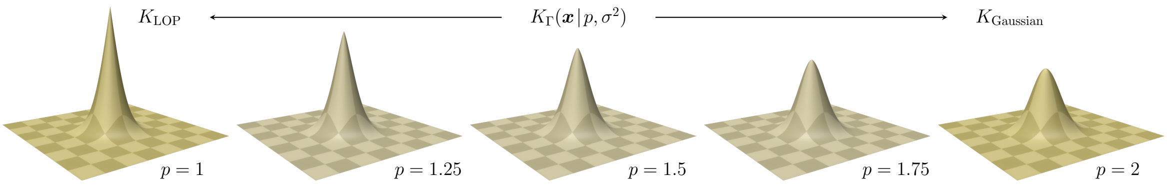


Figure 4.2: Interpolation between 2D incomplete gamma kernels K_Γ with varying $p \in [1, 2]$ and fixed $\sigma^2 = 1/32$. Each kernel corresponds to a localized attraction energy minimization with the respective p-norm.

4.4.3 Generalization via Incomplete Gamma Kernels

Although Mean Shift and Locally Optimal Projection were developed to solve different problems, we can link both concepts by rewriting the update step:

$$\boldsymbol{q}_j^{(t+1)} = \boldsymbol{q}_j^{(t)} + \boldsymbol{m}_{\mathcal{P},G_{\mathrm{LOP}}}(\boldsymbol{q}_j^{(t)}) - \mu\, \boldsymbol{m}_{\boldsymbol{Q}_j^{(t)},G_{\mathrm{rep}}}(\boldsymbol{q}_j^{(t)}) \tag{4.8}$$

This reveals that the LOP operator is a combination of standard Mean Shift with respect to the target set $\mathcal{P}$ and reverse Blurring Mean Shift with respect to the moving source set $\boldsymbol{Q}_j^{(t)} = \boldsymbol{Q}^{(t)} \setminus \{\boldsymbol{q}_j^{(t)}\}$. Therefore, we can interpret the localized L_1 attraction energy minimization with a Gaussian kernel as a global density maximization with respect to a different kernel K_{LOP}. An example of this relation is shown in Figure 4.1.

To derive this corresponding kernel, we consider the profile of the involved kernel which follows a gamma distribution $f_\Gamma(x\,|\,a, b) \propto x^{a-1}\,\mathrm{e}^{-x/b}$ with support $x \in (0, \infty)$ and parameters $a > 0, b > 0$. The profile of the actual kernel then follows the distribution $\bar{F}_\Gamma(x\,|\,a, b) \propto \Gamma(a, x/b)$ which is the complementary CDF of f_Γ and based on the *upper incomplete gamma function* $\Gamma(a, x) = \int_x^\infty t^{a-1}\,\mathrm{e}^{-t}\,\mathrm{d}t$. Since we want to define a d-dimensional kernel K_Γ, we also need to compute the respective normalization constant. For this, we switch the integration domain to d-dimensional spherical coordinates and substitute $s = r^2/b$:

$$\begin{aligned} \frac{1}{c_{K_\Gamma}} &= \int_{\mathbb{R}^d} \Gamma(a, \tfrac{\|\boldsymbol{x}\|^2}{b})\,\mathrm{d}\boldsymbol{x} = \int_\Omega \int_0^\infty \Gamma(a, \tfrac{r^2}{b})\, r^{d-1}\,\mathrm{d}r\,\mathrm{d}\boldsymbol{\Omega} \\ &= \frac{b^{\frac{d}{2}}}{2} \left[\int_\Omega \mathrm{d}\boldsymbol{\Omega}\right] \left[\int_0^\infty \Gamma(a, s)\, s^{\frac{d}{2}-1}\,\mathrm{d}s\right] \end{aligned} \tag{4.9}$$

Due to radial symmetry, both integrals can be solved independently. The former one describes the surface area of the d-dimensional unit sphere and has the closed form $\int_\Omega \mathrm{d}\boldsymbol{\Omega} = 2\pi^{d/2}/\Gamma(d/2)$. Using the relation $\int_0^\infty \Gamma(a, x)\, x^{b-1}\,\mathrm{d}x = \Gamma(a + b)/b$ [Bateman, 1953], we get an expression for the latter one in terms of the ordinary gamma function. We can also apply the recursive relation of the gamma function $\Gamma(a + 1) = a\,\Gamma(a)$ and conclude that $1/c_{K_\Gamma} = (\pi b)^{d/2}\,\Gamma(d/2 + a)/\Gamma(d/2 + 1)$. Finally, we change the parametrization by setting

Table 4.1: Properties of Incomplete Gamma Kernels $K_\Gamma(\boldsymbol{x}\,|\,p,\sigma^2)$ in $\mathbb{R}^d$ for $p > 0$

Distribution	Mean	$\mathbf{0}$
	Covariance	$\frac{d+p}{d+2}\,\sigma^2\,\mathbf{I}$
	Characteristic function	${}_1\mathrm{F}_1(\frac{d+p}{2}, \frac{d+2}{2}, -\frac{\sigma^2\lVert\boldsymbol{\omega}\rVert^2}{2})$
	Moment-generating function	${}_1\mathrm{F}_1(\frac{d+p}{2}, \frac{d+2}{2}, \frac{\sigma^2\lVert\boldsymbol{\omega}\rVert^2}{2})$
Mean Shift	Differentiable profile	✓ except for $x = 0$ if $p \in (0,2)$
	Strictly decreasing profile	✓
	Convex profile	✓ for $p \in (0,2]$
	Analytic	✓
	Bounded	✓
Other	Completely monotonic profile	✓ for $p \in (0,2]$
	Strictly positive definite	✓ for $p \in (0,2]$

$a = p/2, b = 2\sigma^2$ to obtain the final kernel:

$$\boxed{K_\Gamma(\boldsymbol{x}\,|\,p,\sigma^2) = \frac{1}{(2\pi\sigma^2)^{\frac{d}{2}}}\frac{\Gamma(\frac{d+2}{2})}{\Gamma(\frac{d+p}{2})}\,\Gamma(\tfrac{p}{2}, \tfrac{\lVert\boldsymbol{x}\rVert^2}{2\sigma^2})} \tag{4.10}$$

These *incomplete gamma kernels* span a family of Mean Shift kernels corresponding to L_p estimators of the attraction energy localized by a Gaussian kernel. An important special case of this family is the LOP kernel for which we choose $p = 1, \sigma^2 = 1/32$ and apply the identity $\Gamma(1/2, x) = \sqrt{\pi}\ \mathrm{erfc}(\sqrt{x})$ to get

$$K_{\mathrm{LOP}}(\boldsymbol{x}) = \frac{4^d}{\pi^{\frac{d-1}{2}}}\frac{\Gamma(\frac{d+2}{2})}{\Gamma(\frac{d+1}{2})}\ \mathrm{erfc}(4\,\lVert\boldsymbol{x}\rVert) \tag{4.11}$$

where erfc denotes the *complementary error function*. Another special case is the corresponding Gaussian kernel K_{Gaussian} obtained by setting $p = 2$ which is a common choice in Mean Shift and has been extensively analyzed as the localized L_2 estimator of the geometric mean. Figure 4.2 shows an interpolation between these kernels by varying the p-norm.

4.5 Kernel Properties

In the following, we derive several theoretical properties of the family of incomplete gamma kernels which are summarized in Table 4.1.

4.5.1 Characteristic Function and Fourier Transform

In order to gain a deeper understanding of the proposed kernel family K_Γ, we are interested in its characteristic function φ_Γ which can also be interpreted as the Fourier transform $\mathcal{F}$. First, we can apply the relation between the d-dimensional Fourier transform of a radially symmetric function $f(\boldsymbol{x})$ in terms of the Hankel transform of order $d/2-1$ of the function $\|\boldsymbol{x}\|^{d/2-1} f(\boldsymbol{x})$ [Stein and Weiss, 1971] to reduce the dimensionality of the integral to the radial component

$$\begin{aligned}\varphi_\Gamma(\boldsymbol{\omega}) &= \mathcal{F}\left[K_\Gamma(\boldsymbol{x}\,|\,p,\sigma^2)\right](\boldsymbol{\omega}) = \int_{\mathbb{R}^d} K_\Gamma(\boldsymbol{x}\,|\,p,\sigma^2)\,\mathrm{e}^{\mathrm{i}\langle\boldsymbol{\omega}|\boldsymbol{x}\rangle}\,\mathrm{d}\boldsymbol{x} \\ &= c_{K_\Gamma}\frac{(2\pi)^{\frac{d}{2}}}{\|\boldsymbol{\omega}\|^{\frac{d}{2}-1}}\int_0^\infty \Gamma(\tfrac{p}{2},\tfrac{r^2}{2\sigma^2})\,\mathrm{J}_{\frac{d}{2}-1}(\|\boldsymbol{\omega}\|\,r)\,r^{\frac{d}{2}}\,\mathrm{d}r\end{aligned} \tag{4.12}$$

where $\mathrm{J}_q(x)$ denotes the *Bessel function of the first kind* of order q. This integral has the closed-form solution (see the Appendix for a more detailed derivation):

$$\begin{aligned}&c_{K_\Gamma}\frac{(2\pi)^{\frac{d}{2}}}{\|\boldsymbol{\omega}\|^{\frac{d}{2}-1}}\int_0^\infty \Gamma(\tfrac{p}{2},\tfrac{r^2}{2\sigma^2})\,\mathrm{J}_{\frac{d}{2}-1}(\|\boldsymbol{\omega}\|\,r)\,r^{\frac{d}{2}}\,\mathrm{d}r \\ &= c_{K_\Gamma}(2\pi\sigma^2)^{\frac{d}{2}}\frac{\Gamma(\frac{d+p}{2})}{\Gamma(\frac{d+2}{2})}\,{}_1\mathrm{F}_1(\tfrac{d+p}{2},\tfrac{d+2}{2},-\tfrac{\sigma^2\|\boldsymbol{\omega}\|^2}{2}) \\ &= {}_1\mathrm{F}_1(\tfrac{d+p}{2},\tfrac{d+2}{2},-\tfrac{\sigma^2\|\boldsymbol{\omega}\|^2}{2})\end{aligned} \tag{4.13}$$

Therefore, the characteristic function of the incomplete gamma kernel can be written in terms of the *confluent hypergeometric function of the first kind* ${}_1\mathrm{F}_1$. Figure 4.3 shows a comparison between the Gaussian kernel ($p = 2$) and the LOP kernel ($p = 1$) both in spatial and in frequency domain.

If we consider the special case ${}_1\mathrm{F}_1(a,a,x) = \mathrm{e}^x$, we can observe that this result is consistent with the Fourier transform of the Gaussian kernel. Furthermore as $d \to \infty$, the entire family of localized L_p kernel estimators converges to the L_2 estimator since distances become increasingly similar in higher dimensions due to the curse of dimensionality.

4.5.2 Moment-generating Function

Another closely related and useful quantity to consider is the moment-generating function M_Γ of the kernel K_Γ which can be used to compute the mean $\boldsymbol{\mu}_\Gamma$ and covariance matrix $\boldsymbol{\Sigma}_\Gamma$. Although there is a direct connection to the characteristic function in terms of

$$M_\Gamma(\boldsymbol{\omega}) = \varphi_\Gamma(-\mathrm{i}\,\boldsymbol{\omega}) = {}_1\mathrm{F}_1(\tfrac{d+p}{2},\tfrac{d+2}{2},\tfrac{\sigma^2\|\boldsymbol{\omega}\|^2}{2}) \tag{4.14}$$

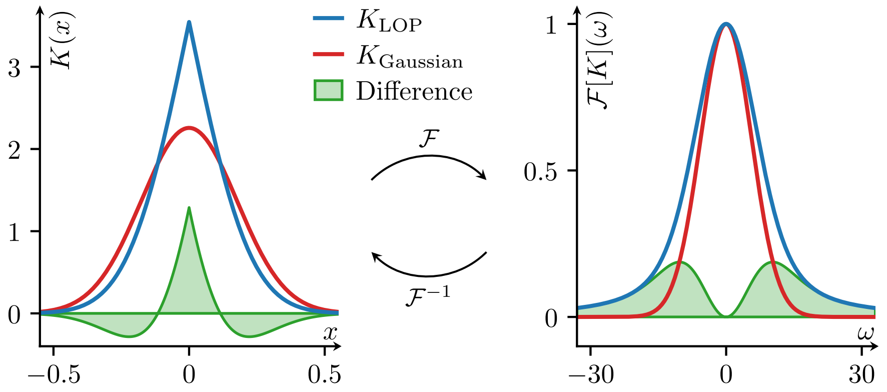


Figure 4.3: Comparison of LOP and Gaussian kernels in spatial and frequency domain. Filtering with the LOP kernel K_{LOP} better preserves higher frequency information.

it does not necessarily exist in general, so we have to prove this property for all $\boldsymbol{\omega} \in \mathbb{R}^d$. For this purpose, we repeatedly apply the comparison theorem of calculus to derive a finite upper bound of the integral. After switching to spherical coordinates, we bound $\cos(\angle(\boldsymbol{\omega}, \boldsymbol{x})) \leq 1$ to decouple the radial component from the angular one:

$$\begin{aligned} M_\Gamma(\boldsymbol{\omega}) &= \int_{\mathbb{R}^d} K_\Gamma(\boldsymbol{x} \,|\, p, \sigma^2)\, \mathrm{e}^{\langle \boldsymbol{\omega} | \boldsymbol{x} \rangle}\, \mathrm{d}\boldsymbol{x} \\ &= c_{K_\Gamma} \int_\Omega \int_0^\infty \Gamma\big(\tfrac{p}{2}, \tfrac{r^2}{2\sigma^2}\big)\, r^{d-1}\, \mathrm{e}^{\|\boldsymbol{\omega}\| r \cos(\angle(\boldsymbol{\omega}, \boldsymbol{x}))}\, \mathrm{d}r\, \mathrm{d}\boldsymbol{\Omega} \\ &\leq c_1 \int_0^\infty \Gamma\big(\tfrac{p}{2}, \tfrac{r^2}{2\sigma^2}\big)\, r^{d-1}\, \mathrm{e}^{\|\boldsymbol{\omega}\| r}\, \mathrm{d}r \end{aligned} \tag{4.15}$$

For brevity, we put finite terms into constants c_i. Next, we combine the two individual upper bounds

$$\Gamma(a, x) \leq \begin{cases} a\, x^{a-1}\, \mathrm{e}^{-x}, & a \in [1, \infty), x \in [a, \infty) \text{ [Natalini and Palumbo, 2000]} \\ x^{a-1}\, \mathrm{e}^{-x}, & a \in (0, 1], x \in (0, \infty) \text{ (partial int.)} \end{cases} \tag{4.16}$$

and apply them after splitting the integral at $r_0 = \max(1, p/2)$. Furthermore, we simplify the expression by completing the square in the exponential term:

$$\begin{aligned} & c_1 \int_0^\infty \Gamma\big(\tfrac{p}{2}, \tfrac{r^2}{2\sigma^2}\big)\, r^{d-1}\, \mathrm{e}^{\|\boldsymbol{\omega}\| r}\, \mathrm{d}r \\ \leq\; & c_1\, c_2 + c_1 \int_{r_0}^\infty r_0 \big(\tfrac{r^2}{2\sigma^2}\big)^{\frac{p}{2}-1}\, \mathrm{e}^{-\frac{r^2}{2\sigma^2}}\, r^{d-1}\, \mathrm{e}^{\|\boldsymbol{\omega}\| r}\, \mathrm{d}r \\ =\; & c_1\, c_2 + c_1\, c_3 \int_{r_0}^\infty r^{p+d-3}\, \mathrm{e}^{-\frac{(r - \sigma^2 \|\boldsymbol{\omega}\|)^2}{2\sigma^2}}\, \mathrm{d}r \end{aligned} \tag{4.17}$$

Since $r \in [1, \infty)$, the remaining polynomial can be bound by a higher order $k = \max(0, \lceil p + d - 3 \rceil) \in \mathbb{N}_0$. Therefore, this integral describes the (incomplete) k-th raw moment of a

1D normal distribution and is finite for any $k \in \mathbb{N}_0$ which, in turn, proves that $M_\Gamma(\boldsymbol{\omega}) < \infty$ exists.

4.5.2.1 Mean

We can now use the moment-generating function M_Γ to directly compute all raw moments of the kernel K_Γ by evaluating the respective derivative at $\boldsymbol{\omega} = \mathbf{0}$. For the mean, we consider the first-order derivative

$$\frac{\partial}{\partial \boldsymbol{\omega}} M_\Gamma(\boldsymbol{\omega}) = \tfrac{d+p}{d+2}\,\sigma^2\,{}_1\mathrm{F}_1(\tfrac{d+p+2}{2}, \tfrac{d+4}{2}, \tfrac{\sigma^2\|\boldsymbol{\omega}\|^2}{2})\,\boldsymbol{\omega} \tag{4.18}$$

and get $\boldsymbol{\mu}_\Gamma = \frac{\partial}{\partial \boldsymbol{\omega}} M_\Gamma(\mathbf{0}) = \mathbf{0}$ as the expected result for a radially symmetric kernel.

4.5.2.2 Covariance

Similarly, we compute the second-order derivative

$$\begin{aligned}\frac{\partial^2}{\partial \boldsymbol{\omega}\,\partial \boldsymbol{\omega}^\mathsf{T}} M_\Gamma(\boldsymbol{\omega}) = \tfrac{d+p}{d+2}\,\sigma^2 \Big[&{}_1\mathrm{F}_1(\tfrac{d+p+2}{2}, \tfrac{d+4}{2}, \tfrac{\sigma^2\|\boldsymbol{\omega}\|^2}{2})\,\mathbf{I} \\ &+ \tfrac{d+p+2}{d+4}\,\sigma^2\,{}_1\mathrm{F}_1(\tfrac{d+p+4}{2}, \tfrac{d+6}{2}, \tfrac{\sigma^2\|\boldsymbol{\omega}\|^2}{2})\,\boldsymbol{\omega}\boldsymbol{\omega}^\mathsf{T}\Big]\end{aligned} \tag{4.19}$$

and obtain the covariance matrix of the kernel K_Γ using the first-order and second-order raw moments as $\boldsymbol{\Sigma}_\Gamma = \frac{\partial^2}{\partial \boldsymbol{\omega}\,\partial \boldsymbol{\omega}^\mathsf{T}} M_\Gamma(\mathbf{0}) - \boldsymbol{\mu}_\Gamma \boldsymbol{\mu}_\Gamma^\mathsf{T} = (d+p)/(d+2)\,\sigma^2\,\mathbf{I}$.

4.5.3 Mean Shift Properties

In addition to the distribution-specific properties above, we can get further insights into the kernel family by exploiting the comprehensive theory that has been developed for the Mean Shift algorithm [Yamasaki and Tanaka, 2019]. This requires proving several additional properties including that the kernel K_Γ is bounded and analytic and that its profile k_Γ is differentiable, strictly decreasing, and convex.

4.5.3.1 Differentiability, Monotonicity and Convexity

In order to show that the profile is strictly decreasing, we consider its first-order derivative

$$\frac{\mathrm{d}}{\mathrm{d}x} k_\Gamma(x \,|\, p, \sigma^2) = \frac{\mathrm{d}}{\mathrm{d}x} \Gamma(\tfrac{p}{2}, \tfrac{x}{2\sigma^2}) = -(\tfrac{1}{2\sigma^2})^{\frac{p}{2}}\, x^{\frac{p}{2}-1}\, \mathrm{e}^{-\frac{x}{2\sigma^2}} \tag{4.20}$$

which is defined for all $x \in (0, \infty)$ as well as for $x = 0$ if $p \in [2, \infty)$. Since the involved polynomial and exponential terms are always positive, it follows that the derivative must be negative, that is $\frac{\mathrm{d}}{\mathrm{d}x} k_\Gamma(x \,|\, p, \sigma^2) < 0$, and the profile strictly decreasing.

Similarly, we see that the second-order derivative is given by

$$\frac{\mathrm{d}^2}{\mathrm{d}x^2}k_\Gamma(x\,|\,p,\sigma^2) = \frac{\mathrm{d}}{\mathrm{d}x}k_\Gamma(x\,|\,p,\sigma^2)\left[\frac{\frac{p}{2}-1}{x} - \frac{1}{2\sigma^2}\right] \tag{4.21}$$

where, in order to ensure that $\frac{\mathrm{d}^2}{\mathrm{d}x^2}k_\Gamma(x\,|\,p,\sigma^2) > 0$, the latter term must be non-positive which is equivalent to the condition $x \geq (p-2)\,\sigma^2$. Since this should hold for all $x \in (0,\infty)$, convexity is only guaranteed for kernels with $p \in (0,2]$ which, in particular, includes the Gaussian kernel ($p = 2$) as well as the LOP kernel ($p = 1$).

4.5.3.2 Boundedness and Analyticity

Instead of showing both properties for the kernel K_Γ itself, it is sufficient to show them for its profile k_Γ. Since k_Γ is non-negative and monotonically decreasing, we only have to consider the case $x = 0$. For this value, $\Gamma(a,x)$ reduces to the gamma function $\Gamma(a)$ which is finite for $a > 0$. Furthermore, analyticity directly follows from the fact that $\Gamma(a,x)$ is holomorphic in $x \in (0,\infty)$ for any fixed $a > 0$.

4.5.3.3 Consequences for the LOP operator

The aforementioned properties have several direct implications [Yamasaki and Tanaka, 2019] on the behavior of the LOP operator (with zero repulsion) as well as to Mean Shift applied with the incomplete gamma kernel K_Γ for $p \in (0,2]$. With the exception of the finite set of target points $\mathcal{P}$ where singularities are introduced in the kernel G_Γ, the following properties hold:

Non-zero Gradient. The gradient of the kernel density estimate $\nabla\hat{f}_{\mathcal{P},K_{\mathrm{LOP}}}$ is non-zero outside the convex hull of the target point set $\mathcal{P}$. This implies that all solutions must lie within the convex hull.

Plateau-free Density. In addition to non-zero gradients, the kernel density estimate function on the set $\mathbb{R}^3 \setminus \mathcal{P}$ has no plateaus. Since the set of target points $\mathcal{P}$ is finite, we can extend this property to the full space $\mathbb{R}^3$.

Non-decreasing Density Estimate. Another interesting subset to consider is the improvement ball $\mathcal{I}(\boldsymbol{q}_j^{(t)})$ which denotes a d-dimensional sphere centered at the point $\boldsymbol{q}_j^{(t)} + \boldsymbol{m}_{\mathcal{P},G_{\mathrm{LOP}}}(\boldsymbol{q}_j^{(t)})$ with radius $\|\boldsymbol{m}_{\mathcal{P},G_{\mathrm{LOP}}}(\boldsymbol{q}_j^{(t)})\|$. In case of the LOP operator, it follows that all points $\boldsymbol{x}$ within the improvement ball have non-decreasing kernel density estimates $\hat{f}_{\mathcal{P},K_{\mathrm{LOP}}}(\boldsymbol{x}) \geq \hat{f}_{\mathcal{P},K_{\mathrm{LOP}}}(\boldsymbol{q}_j^{(t)})$.

Convergence of Density Estimate Sequence. As a consequence of the above property, the sequence of kernel density estimates $\{\hat{f}_{\mathcal{P},K_{\mathrm{LOP}}}(\boldsymbol{q}^{(t)})\}$ obtained via the fixed-point iteration $\boldsymbol{q}^{(t+1)} = \boldsymbol{q}^{(t)} + \boldsymbol{m}_{\mathcal{P},G_{\mathrm{LOP}}}(\boldsymbol{q}^{(t)})$ is non-decreasing. Furthermore, this sequence always converges.

Convergence of Mode Estimate Sequence. Finally, we can conclude that the mode estimate sequence $\{\boldsymbol{q}^{(t)}\}$ converges to a single point. Depending on the window size h and the distribution of the target points $\mathcal{P}$, this solution could be either a point $\boldsymbol{p} \in \mathcal{P}$ due to the singularity (for very small window sizes) or a different point $\boldsymbol{p} \in \mathbb{R}^3 \setminus \mathcal{P}$ in the corresponding convex hull (for larger window sizes).

4.5.4 Further Properties

Besides the theoretical results of our proposed kernel family concerning basic distribution-related aspects as well as insights in the behavior and structure of the Mean Shift algorithm, we want to derive further properties that opens up a broader set of applications such as solving linear equation systems.

4.5.4.1 Complete Monotonicity

For this purpose, we show that the kernel profile k_Γ is completely monotonic, that is $(-1)^n \frac{\mathrm{d}^n}{\mathrm{d}x^n} k_\Gamma(x \,|\, p, \sigma^2) \geq 0$ for all $n \in \mathbb{N}_0$ and $x \in (0, \infty)$. From the derivation of the Mean Shift properties, we already know that $k_\Gamma(x \,|\, p, \sigma^2) > 0$ and $\frac{\mathrm{d}}{\mathrm{d}x} k_\Gamma(x \,|\, p, \sigma^2) < 0$ holds for all $x \in (0, \infty)$. Since x^a with $a \leq 0$ as well as e^{-x} are both completely monotonic and the product of two completely monotonic functions retains that property [Schilling et al., 2012], we can conclude that this also holds for $k_\Gamma(x \,|\, p, \sigma^2)$ with $p \in (0, 2]$.

4.5.4.2 Strict Positive Definiteness

A direct consequence of the complete monotonicity of its profile is that the kernel K_Γ must be a strictly positive definite function [Schoenberg, 1938]. Therefore, for any set of points $\mathcal{P}$, the matrix

$$\boldsymbol{C} = \big(K_\Gamma(\boldsymbol{p}_i - \boldsymbol{p}_j \,|\, p, \sigma^2)\big)_{ij} \in \mathbb{R}^{|\mathcal{P}| \times |\mathcal{P}|} \tag{4.22}$$

is symmetric and positive definite for $p \in (0, 2]$, so any linear system with respect to $\boldsymbol{C}$ has a unique solution which can be computed by, e.g., conjugate gradient solvers. This also directly extends to any truncated version of K_Γ where the matrix $\boldsymbol{C}$ becomes sparse and more efficient to solve as vanishing derivatives of the truncated profile do not affect complete monotonicity.

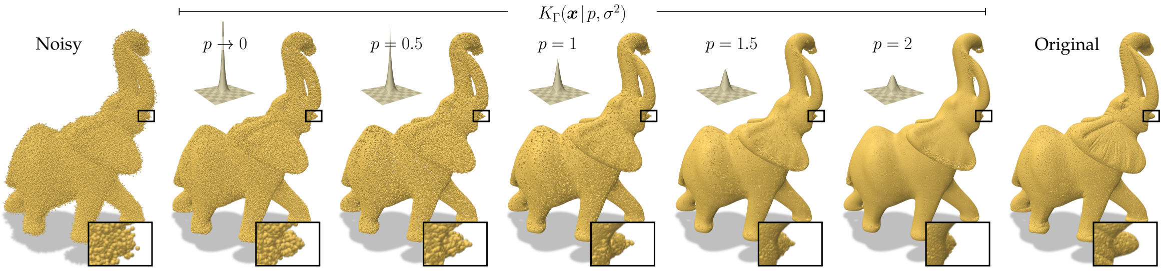


Figure 4.4: Exemplary point cloud denoising of the *Elephant* model (302 458 points) with 30 WLOP [Huang et al., 2009] iterations ($h = 6$, $\mu = 0.4$) for different incomplete gamma kernels K_Γ using varying $p \in (0, 2]$ and fixed $\sigma^2 = 1/32$. The model has been corrupted with $\sigma_{\text{noise}} = 0.3$ (80 % points) and $\sigma_{\text{outlier}} = 1.5$ Gaussian noise (20 % points) respectively to account for both typical sensor noise and heavy outliers. Higher p-norms result in more regular but oversmoothed point distributions whereas lower values better preserve features. Unit of h, σ_{noise}, σ_{outlier}: [% BB diagonal].

4.6 Applications

Besides the application of other localized L_p estimators for point cloud denoising via the incomplete gamma kernels K_Γ, as shown in Figure 4.4, we illustrate several further applications to demonstrate the benefits of the theoretical results derived for the kernel family.

4.6.1 WLOP Density Weights

Although the repulsion term mitigates the clustering effect of the attraction term, the projection is still highly dependent on the distribution of the target points $\mathcal{P}$. This has been addressed in WLOP [Huang et al., 2009] by computing weights v_i for each target point $\boldsymbol{p}_i$ and $v_j^{(t)}$ for each projection point $\boldsymbol{q}_j^{(t)}$ based on the reciprocal and ordinary density value respectively with the (unnormalized) localization kernel θ. However, our derived theoretical properties reveal two major limitations of this particular choice: 1) Although the Gaussian localization kernel θ could be considered a reasonable approximation of the actual kernel K_{LOP} (see Figure 4.3), high-frequency information in the density estimate is not properly handled and smoothed out; 2) taking the reciprocal to invert the density of $\boldsymbol{p}_i$ ignores the dependencies between the weights which corresponds to the assumption of constant density in a window of size h. In order to achieve a more accurate normalization, we propose two novel weighting schemes.

Simple Scheme. A simple extension to the WLOP weights keeps the assumption of the latter limitation and addresses only the former one by applying the actual kernel K_{LOP}, that is estimating the weights

$$v_i = \frac{1}{\hat{f}_{\mathcal{P}, K_{\text{LOP}}}(\boldsymbol{p}_i)}, \quad v_j^{(t)} = \hat{f}_{\mathcal{Q}^{(t)}, K_{\text{LOP}}}(\boldsymbol{q}_j^{(t)}) \tag{4.23}$$

Table 4.2: Parameter Sets of Kernel Approximation $\hat{K}_{\text{LOP}}$

	CLOP [Preiner et al., 2014]		Ours		Ours (Consistent)	
k	$\hat{w}_k$	$\hat{\sigma}_k$	$\hat{w}_k$	$\hat{\sigma}_k$	$\hat{w}_k$	$\hat{\sigma}_k$
1	97.761	0.01010	61.509	0.02102	46.409	0.03118
2	29.886	0.03287	11.932	0.07289	9.635	0.10582
3	11.453	0.11772	5.069	0.15700	2.674	$\sqrt{1/32}$

via the density of the point clouds $\mathcal{P}$ and $\boldsymbol{Q}^{(t)}$ respectively. This scheme can be easily integrated into existing applications of WLOP as it only involves a different kernel function in the overall weight computation.

Full Scheme. To address both limitations, we consider the kernel density estimate function $\hat{f}_{\mathcal{P},K_{\text{LOP}}}$ with weights v_i applied to each term. We want to enforce constant density at the points $\boldsymbol{p}_i$ which can be formulated as a linear optimization problem in matrix form:

$$\frac{1}{|\mathcal{P}|\,h^d}\left(K_{\text{LOP}}(\tfrac{\boldsymbol{p}_i-\boldsymbol{p}_j}{h})\right)_{ij}\boldsymbol{v} = \mathbf{1} \tag{4.24}$$

This corresponds to radial basis function (RBF) interpolation and we can obtain a unique solution since K_{LOP} is strictly positive definite. Furthermore, we truncate the kernel at $h/2$ to drastically reduce the memory requirements of the matrix and use a sparse conjugate gradient solver. In case of the projection points $\boldsymbol{q}_j^{(t)}$, we consider the inverse of the matrix which leads to the same weights as in the simple scheme.

4.6.2 CLOP Kernel Approximation

Both LOP and Mean Shift operate on a discrete set of points and are formulated as discrete sums over the point cloud which may have a significant runtime cost on large datasets. To allow for a more compact modeling of the input data, CLOP [Preiner et al., 2014] replaces the point set $\mathcal{P}$ by a smaller set of normal distributions $\mathcal{P}_{\mathcal{N}} = \{(w_i, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)\}$ with weights w_i, means $\boldsymbol{\mu}_i$, and local covariance matrices $\boldsymbol{\Sigma}_i$ and extends the attraction energy to the continuous space. However, since the integral in the respective update step cannot be directly solved, the kernel $\alpha(\|\boldsymbol{x}\|)$ is approximated by a radially symmetric Gaussian mixture model $\hat{\alpha}(\boldsymbol{x}) = 1/h\sum_{k=1}^{3}\hat{w}_k\,\hat{c}_k\,\mathcal{N}(\boldsymbol{x}/h\,|\,\mathbf{0},\hat{\sigma}_k^2\,\mathbf{I})$ consisting of three components with fitted parameters $\{(\hat{w}_k,\hat{\sigma}_k)\}$ and dimension-dependent constants $\hat{c}_k = |2\pi\hat{\sigma}_k^2\,\mathbf{I}|^{1/2}$. In the context of Mean Shift, this implies that the kernel G_{LOP} is in fact approximated which directly allows us to derive

$$\hat{K}_{\text{LOP}}(\boldsymbol{x}) = \frac{\sum_{k=1}^{3}\hat{\sigma}_k^2\,\hat{w}_k\,\hat{c}_k\,\mathcal{N}(\boldsymbol{x}\,|\,\mathbf{0},\hat{\sigma}_k^2\,\mathbf{I})}{\sum_{k=1}^{3}\hat{\sigma}_k^2\,\hat{w}_k\,\hat{c}_k} \tag{4.25}$$

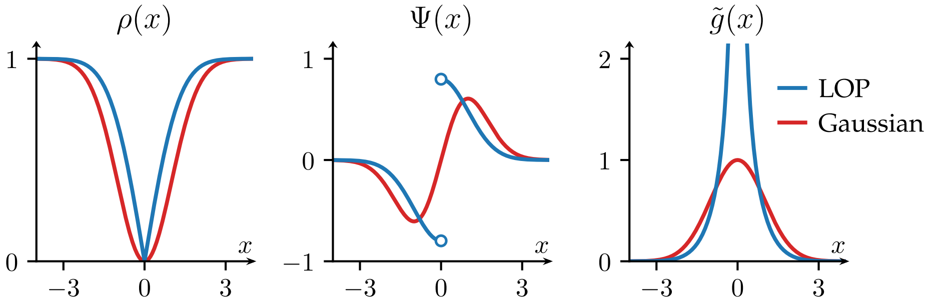


Figure 4.5: Comparison of LOP and Gaussian M-estimators for $\sigma^2 = 1$. Due to the close relation to Mean Shift, the robust loss functions ρ do not only share the shape of the corresponding kernels K but also have similar properties.

as an approximation of the kernel K_{LOP} with the same set of fitted parameters $\{(\hat{w}_k, \hat{\sigma}_k)\}$.

Kernel Fit. However, finding the optimal parameter set is highly challenging due to the singularity of G_{LOP} at $x = 0$ and, thereby, the unbounded ratio between the smallest and largest sampling value in the half-open fitting interval $(0, 1]$ for $h = 1$. In contrast, we directly optimize on the kernel K_{LOP} which does not suffer from these limitations. We fix the parameter $w_3 = 1$ to constrain the remaining degree of freedom and obtain the solution from 10^7 uniformly sampled points in the interval $[0, 1]$ via the Levenberg-Marquardt algorithm (see Table 4.2). Although the LOP operator is scale-invariant in terms of the kernel α, we nevertheless estimate a global scaling factor for the weights $\hat{w}_k$ via Levenberg-Marquardt optimization in the interval $(0.01, 1]$ for a better comparability with CLOP.

Consistent Fit. We can also see from the definition of the kernel approximation $\hat{K}_{\text{LOP}}$ that its variance consists of a convex combination of the individual variances $\hat{\sigma}_k^2$:

$$\hat{\sigma}^2_{\text{LOP}} = \frac{\sum_{k=1}^{3} \hat{w}_k \, \hat{\sigma}_k^{d+4}}{\sum_{k=1}^{3} \hat{w}_k \, \hat{\sigma}_k^{d+2}} \tag{4.26}$$

In the limit $d \longrightarrow \infty$, this combination degenerates to $\hat{\sigma}^2_{\text{LOP}} \longrightarrow \max_k \hat{\sigma}_k^2$ which is similar to the maximum norm L_∞ being the limit of the L_p norms. Therefore, we can enforce an additional consistency constraint in the parameter optimization process by fixing the parameter $\hat{\sigma}_3 = \sqrt{1/32}$ to match the expected standard deviation.

4.6.3 Robust Loss Functions

In the context of point cloud reconstruction, LOP formulates the projection onto the underlying surface via the localized L_1 median in a robust way. Similarly, mesh denoising applications aim to improve the quality of meshes in a two-stage approach where the face normals are initially filtered and subsequently used in the second stage to estimate the original vertex positions. Obtaining a reliable estimate of a surface normal $\boldsymbol{n}$ can be performed by M-estimators in the field of robust statistics which is also related to the concept of anisotropic diffusion [Black et al., 1998]. Here, the objective function

$$L(\boldsymbol{n}) = \sum_i \rho(\|\boldsymbol{n}_i - \boldsymbol{n}\|) \tag{4.27}$$

defined with a robust loss function ρ is considered and a solution can be found based on the corresponding influence function $\Psi(x) = \frac{\mathrm{d}}{\mathrm{d}x}\rho(x)$ and anisotropic weight function $\tilde{g}(x) = \Psi(x)/x$ [Yadav et al., 2021]:

$$\boldsymbol{n}^{(t+1)} = \frac{\sum_i \tilde{g}(\|\boldsymbol{n}_i - \boldsymbol{n}^{(t)}\|)\, \boldsymbol{n}^{(t)}}{\left\|\sum_i \tilde{g}(\|\boldsymbol{n}_i - \boldsymbol{n}^{(t)}\|)\, \boldsymbol{n}^{(t)}\right\|} \tag{4.28}$$

This result is closely related to the derivation of Mean Shift and shares many properties with it [Comaniciu and Meer, 2002]. Thus, we can define the family of *incomplete gamma losses* along with the respective influence and anisotropic weight functions:

$$\rho_\Gamma(x \,|\, p, \sigma^2) = \frac{1}{\Gamma(\frac{p}{2})}\, \gamma(\tfrac{p}{2}, \tfrac{x^2}{2\sigma^2}) \tag{4.29}$$

$$\Psi_\Gamma(x \,|\, p, \sigma^2) = \frac{2}{(2\sigma^2)^{\frac{p}{2}}\, \Gamma(\frac{p}{2})}\, |x|^{p-2}\, \mathrm{e}^{-\frac{x^2}{2\sigma^2}}\, x \tag{4.30}$$

$$\tilde{g}_\Gamma(x \,|\, p, \sigma^2) = \frac{2}{(2\sigma^2)^{\frac{p}{2}}\, \Gamma(\frac{p}{2})}\, |x|^{p-2}\, \mathrm{e}^{-\frac{x^2}{2\sigma^2}} \tag{4.31}$$

Here, the losses ρ_Γ are built upon the *lower incomplete gamma function* $\gamma(a, x) = \int_0^x t^{a-1}\, \mathrm{e}^{-t}\, \mathrm{d}t$ which is connected to the upper incomplete gamma function via the relation $\gamma(a, x) + \Gamma(a, x) = \Gamma(a)$. By choosing $p = 1$ and applying the identity $\gamma(1/2, x) = \sqrt{\pi}\ \mathrm{erf}(\sqrt{x})$, we get the LOP loss

$$\rho_{\mathrm{LOP}}(x \,|\, \sigma^2) = \mathrm{erf}(\tfrac{|x|}{\sqrt{2\sigma^2}}) \tag{4.32}$$

where erf denotes the *error function* and is related to its complementary counterpart via $\mathrm{erfc}(x) = 1 - \mathrm{erf}(x)$. Considering $\sigma^2 = 1/32$, the relation $\tilde{g}_{\mathrm{LOP}}(x \,|\, 1/32) \propto g_{\mathrm{LOP}}(x^2)$ further highlights the close similarity to Mean Shift. Figure 4.5 shows a comparison between the Gaussian M-estimator ($p = 2$) and the LOP M-estimator ($p = 1$).

4.7 Experimental Results

In the following, we demonstrate the effectiveness of our proposed extensions that are derived from the theoretical properties of the kernel family.

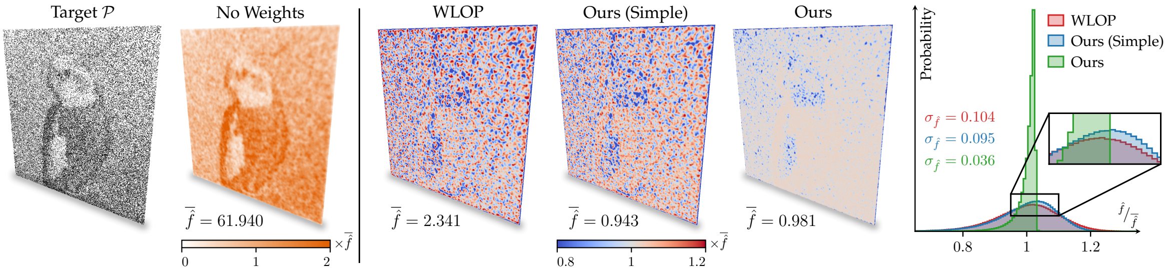


Figure 4.6: Kernel density estimate $\hat{f}$ for $h = 3$ of a planar target surface patch $\mathcal{P}$ (74 000 points) that is sampled inversely proportional to the intensity of the *Bird* image. The regularity σ_Q of any projected point set Q onto this target directly depends on the uniformity of $\hat{f}$. Whereas WLOP [Huang et al., 2009] and our simple weighting scheme cannot fully remove high-frequency variations, our full weighting scheme leads to a significantly better normalization and more uniform density. Unit of h: [% BB diagonal].

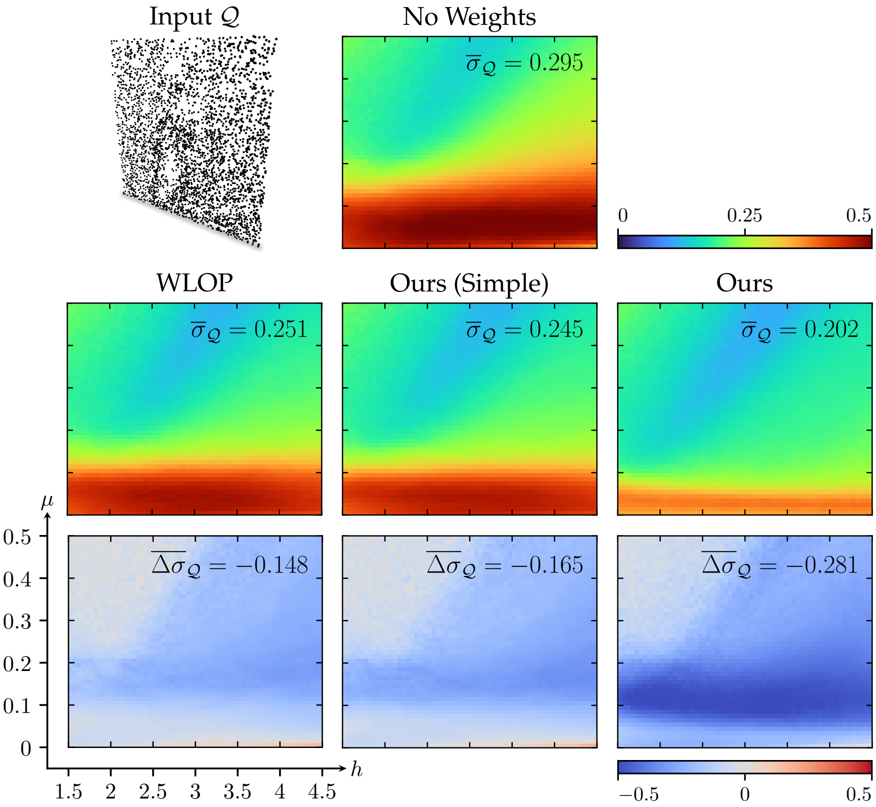


Figure 4.7: Regularity σ_Q of the input subset Q (3 700 points) projected on a planar target surface patch $\mathcal{P}$ (74 000 points) that is sampled inversely proportional to the intensity of the *Bird* image. Due to the better density normalization, our weighting schemes further improve the regularity across various combinations of the window size h and the repulsion weight μ. Unit of h and σ_Q: [% BB diagonal].

4.7.1 Evaluation of WLOP Density Weights

In order to evaluate the performance of our density weighting schemes, we measured the regularity of the point cloud $\boldsymbol{Q}$ after projection onto a highly irregular target $\mathcal{P}$ [Huang et al., 2009]. For this purpose, we sampled 74 000 target points from a 3D surface patch inversely proportional to the intensity of the mapped *Bird* image and took a random subset of 3 700 points for projection. Then, we applied 100 iterations of the LOP operator as well as its weighted versions and computed the regularity

$$\sigma_{\boldsymbol{Q}} = \left[\frac{1}{|\boldsymbol{Q}|} \sum_{i} \Big(d(\boldsymbol{q}_i, \boldsymbol{Q} \setminus \{\boldsymbol{q}_i\}) - \overline{d(\boldsymbol{q}_i, \boldsymbol{Q} \setminus \{\boldsymbol{q}_i\})} \Big)^2 \right]^{\frac{1}{2}} \tag{4.33}$$

which is defined as the standard deviation of the nearest neighbor distances $d(\boldsymbol{x}, \boldsymbol{\mathcal{Y}}) = \min_j \left\| \boldsymbol{x} - \boldsymbol{y}_j \right\|$ within the point cloud $\boldsymbol{Q}$. Figure 4.7 shows the quantitative results for 60×50 combinations of h and μ. Throughout 76.7 % of all combinations, our simple weighting scheme performs better than WLOP with a slightly lower value of $\sigma_{\boldsymbol{Q}}$ on average. Our full scheme outperforms WLOP in 99.3 % and the simple scheme in 98.7 % of all combinations, especially in configurations with low repulsion weights $\mu \in [0, 0.2]$.

These improvements in point cloud regularity directly correspond to a more evenly distributed density along the surface. Figure 4.6 depicts a comparison of 1 000×1 000 evenly sampled density values on the respective 3D surface patch. Both WLOP and our simple scheme normalize the lower frequency components of the density, but still retain high-frequency variations due to the independent computation of each weight. On the other hand, our full scheme does not suffer from these artifacts and only leads to underestimated densities at the boundary and in sparsely sampled regions where the window size h is not sufficiently large to bridge these gaps.

Since Mean Shift and, thereby, LOP and its variants are scale-invariant with respect to a global normalization constant, we computed the mean density $\overline{\hat{f}}$ for each weighting scheme and used this value to normalize each density distribution for a fair comparison. Whereas this value is close to one for both of our schemes due to the correct handling of the normalization constants, we can derive a theoretical estimate of this value for WLOP

$$\overline{\hat{f}}_{\text{WLOP}} \approx \frac{1}{|\mathcal{P}|\, h^3}\, c_{\theta}^{(d=3)}\, \frac{c_{\text{LOP}}^{(d=3)}}{c_{\text{LOP}}^{(d=2)}}\, \frac{c_{\theta}^{(d=2)}}{c_{\theta}^{(d=3)}} = 2.396 \tag{4.34}$$

which consists of three terms: 1) the normalization constant of the kernel density estimate function; 2) the missing normalization constant of the kernel θ; and 3) a dimension-dependent correction factor. The last term models the different domains from which the density is accumulated as we consider a surface patch that corresponds to a 2D subspace embedded in the 3D space. Therefore, the integration domain of the density differs by one dimension which can be accounted for by the ratio of the normalization constants of both the actual density kernel K_{LOP} as well as the chosen kernel θ for density weight computation.

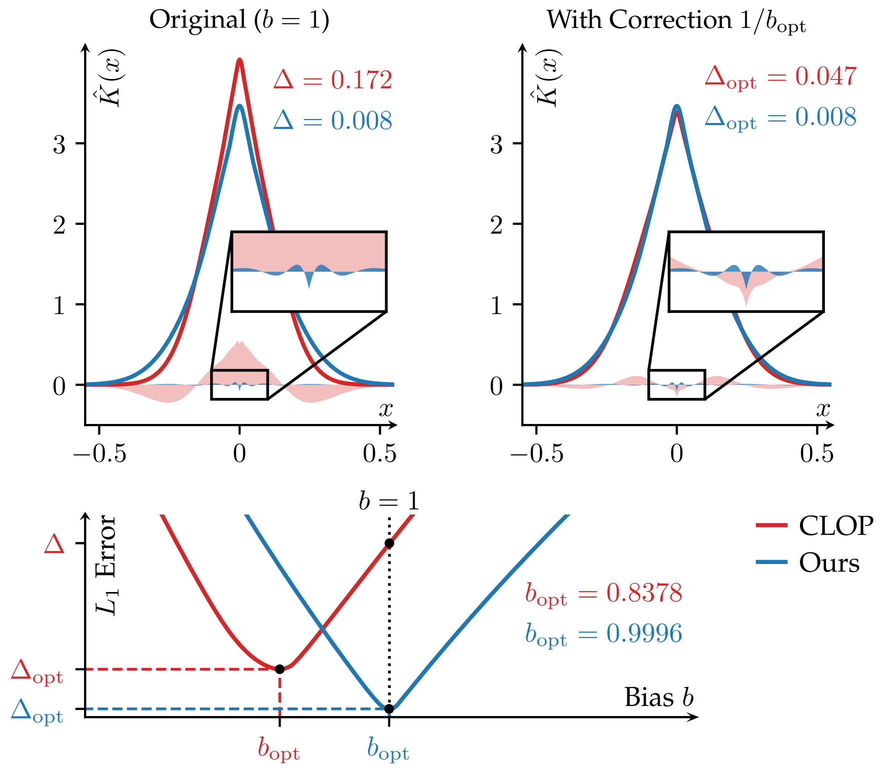


Figure 4.8: Analysis of bias in the width of the kernel approximations $\hat{K}_{\text{LOP}}$ to the original kernel K_{LOP}. A correction by scaling the parameters $\hat{\sigma}_k$ with a global factor $1/b_{\text{opt}}$ lowers the error for CLOP [Preiner et al., 2014]. Nevertheless, our approximation still better follows K_{LOP} with a significantly lower error and is almost unbiased in the 1-dimensional case.

Table 4.3: Ratio of Standard Deviations $\hat{\sigma}_{\text{LOP}}/\sigma_{\text{LOP}}$ Between the Kernel Approximations $\hat{K}_{\text{LOP}}$ and the Original Kernel K_{LOP} in $\mathbb{R}^d$

	$d = 1$	$d = 2$	$d = 3$	$d = 4$	$d \to \infty$
CLOP [Preiner et al., 2014]	0.7931	0.7632	0.7430	0.7291	0.6659
Ours	**0.9917**	**0.9834**	**0.9737**	**0.9640**	0.8881
Ours (Consistent)	1.0137	1.0252	1.0362	1.0440	**1**

4.7.2 Evaluation of CLOP Kernel Approximation

We evaluated the approximation error of our fitted parameter set against the original one proposed by CLOP [Preiner et al., 2014]. First, we quantified systematic errors of the kernel approximation $\hat{K}_{\text{LOP}}$ by analyzing its standard deviation $\hat{\sigma}_{\text{LOP}}$. We can see in Table 4.3 that both CLOP and our approximation underestimate the actual value σ_{LOP} and that the bias increases in higher dimensions. Although these errors are significantly lower for our approximation throughout all dimensions, they may still be significant. Our consistent approximation always overestimates the actual standard deviation and has a slightly higher

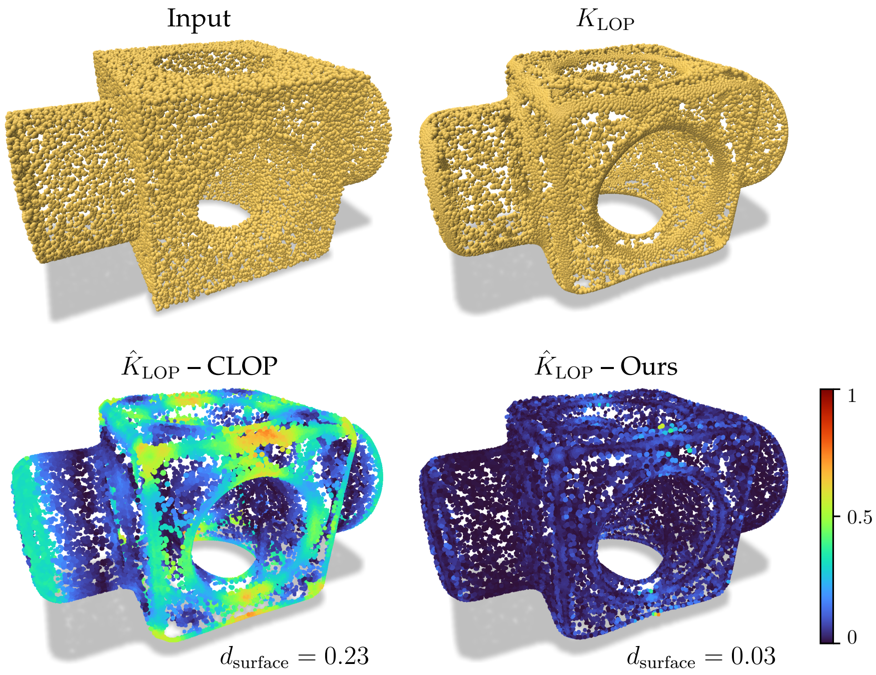

Input
K_{LOP}
$\hat{K}_{\text{LOP}}$ – CLOP
$\hat{K}_{\text{LOP}}$ – Ours
1
0.5
0
$d_{\text{surface}} = 0.23$
$d_{\text{surface}} = 0.03$

Figure 4.9: Bias of the kernel approximations $\hat{K}_{\text{LOP}}$ when applying WLOP [Huang et al., 2009] to smooth the *Block* model (25 000 points) with the window size $h = 25$. Whereas the CLOP [Preiner et al., 2014] approximation introduces systematic errors at the edges due to the bias in the width, our variant closely resembles the behavior of the original kernel K_{LOP}. Unit: [% BB diagonal].

error than the unconstrained variant in low dimensions up to $d = 5$. However, it becomes unbiased in the limit $d \longrightarrow \infty$ and should be preferred in higher dimensions. We also considered minimizing the L_1 distance to the actual kernel K_{LOP} by scaling the values $\hat{\sigma}_k$ with correction factors $1/b_{\text{opt}}$ to obtain an improved set of parameters which is shown in Figure 4.8. Here, the error of our approximation is significantly lower than for CLOP both before and after optimal correction. Furthermore, the optimal scaling factors are similar to the ratios of the standard deviation for $d = 1$.

In addition to the theoretical analysis of the kernel approximations, we also measured the reconstruction error when replacing the actual kernel K_{LOP} with the respective approximation. For this purpose, we chose the *Block* model and uniformly sampled 50 000 target points $\mathcal{P}$ and 25 000 projection points $\mathcal{Q}$ respectively. We applied 100 iterations of WLOP as a smoothing operator with a large window size of $h = 25$ percent of the bounding box diagonal of $\mathcal{P}$ and a repulsion weight $\mu = 0.4$. Then, we measured the distance of each point to the (triangulated) surface of the reference point cloud as well as the mean point-surface distance

$$d_{\text{surface}}(\mathcal{X}, \mathcal{Y}) = \frac{1}{|\mathcal{X}|} \sum_i \min_j d(\boldsymbol{x}_i, t(y_j)) \tag{4.35}$$

where $t(y_j)$ denotes the j-th triangle of $\mathcal{Y}$. Figure 4.9 shows the results of this point cloud smoothing operation. Whereas the CLOP approximation introduces higher errors at the

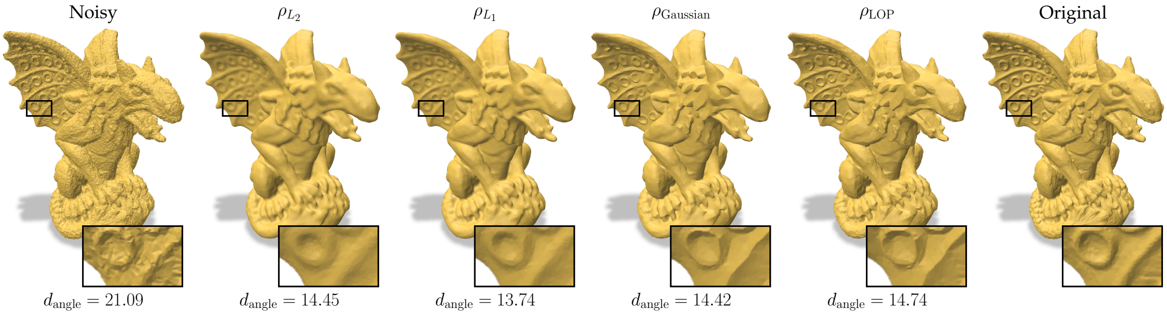

Noisy
ρ_{L_2}
ρ_{L_1}
ρ_{Gaussian}
ρ_{LOP}
Original
$d_{\text{angle}} = 21.09$
$d_{\text{angle}} = 14.45$
$d_{\text{angle}} = 13.74$
$d_{\text{angle}} = 14.42$
$d_{\text{angle}} = 14.74$

Figure 4.10: Mesh denoising of the *Gargoyle* model (86 311 vertices, 172 610 faces) corrupted with $0.25\,\bar{l}_e$ uniform noise. Although the mean angular distance d_{angle} is slightly higher for the LOP loss ρ_{LOP}, features and finer details are better preserved. Unit: [°].

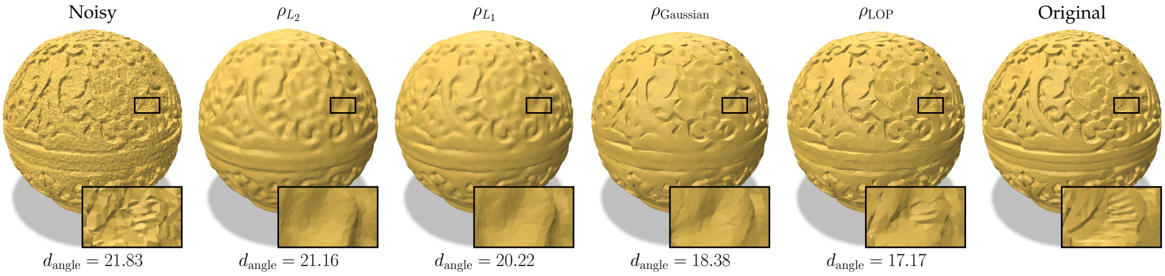

Noisy
ρ_{L_2}
ρ_{L_1}
ρ_{Gaussian}
ρ_{LOP}
Original
$d_{\text{angle}} = 21.83$
$d_{\text{angle}} = 21.16$
$d_{\text{angle}} = 20.22$
$d_{\text{angle}} = 18.38$
$d_{\text{angle}} = 17.17$

Figure 4.11: Mesh denoising of the *Box* model (70 134 vertices, 140 259 faces) corrupted with $0.25\,\bar{l}_e$ uniform noise. Filtering with the LOP loss ρ_{LOP} results in the lowest mean angular distance d_{angle} and reconstructs fine details best. Unit: [°].

edges of the sampled model due to the significantly underestimated standard deviation of the kernel, our approximation does not suffer from these artifacts.

4.7.3 Evaluation of Robust Loss Functions

We tested the LOP M-estimator against other popular choices for normal filtering. For this, we used the *Gargoyle* (86 311 vertices, 172 610 faces) and *Box* (70 134 vertices, 140 259 faces) models and corrupted the vertices in random directions by $0.25\,\bar{l}_e$ uniform noise where $\bar{l}_e$ denotes the average face edge length. Then, we applied 50 iterations of normal filtering with $\sigma = 0.3$ for each face normal $\boldsymbol{n}$ within its geometric neighborhood of size $r = 1.5\,\bar{l}_e$, that is all normals whose face centers are traversable along the surface within a ball of size r. To avoid the singularity of the L_1 and LOP losses at $x = 0$, we only considered the neighboring face normals in the initial iteration and used all normals subsequently. For the second stage of the mesh denoising framework, we used the vertex update by Zhang et al. [2018] with their default parameters of 20 iterations and $w = 0.001$ which avoids the triangle flipping problem. We evaluated the reconstruction error by the mean angular distance

$$d_{\text{angle}}(\mathcal{X}, \mathcal{Y}) = \frac{1}{|\mathcal{X}|} \sum_i \left[\frac{360}{2\pi} \arccos(\langle \boldsymbol{n}(x_i) | \boldsymbol{n}(y_i) \rangle) \right] \tag{4.36}$$

to the face normals of the ground truth mesh. Figures 4.10 and 4.11 show comparisons between the L_2, L_1, Gaussian, and LOP loss. Whereas the L_2 loss leads to a very smooth surface, its L_1 counterpart is less sensitive to large normal variations within the local neighborhood and better preserves features. However, sharp edges cannot be reconstructed since all collected normals are considered in a global fashion. The Gaussian and LOP loss functions can be viewed as localized versions of the former losses and do not suffer from this limitation. Finer details being at a similar scale as the applied noise are hard to reconstruct and mostly smoothed out by all variants, but can be partially recovered by the LOP loss.

4.8 Conclusions

We presented incomplete gamma kernels, a novel family of kernels generalizing LOP operators. By revisiting the classical localized L_1 estimator used in LOP, we revealed its relation to the Mean Shift framework via a novel kernel K_{LOP} and generalized this result to arbitrary localized L_p estimators. We derived several theoretical properties of the kernel family K_Γ concerning distributional, Mean Shift induced, and other aspects such as strict positive definiteness to obtain a deeper understanding of the operator's projection behavior. Furthermore, we illustrated several applications including an improved WLOP density weighting scheme, a more accurate kernel approximation for CLOP, as well as introducing incomplete gamma losses ρ_Γ as a novel set of robust loss functions and confirmed their effectiveness in a variety of quantitative and qualitative experiments. We expect that building upon the insights provided by our work will be beneficial for future developments on surface reconstruction from noisy point clouds.

Chapter 5

SLAMCast: Large-Scale, Real-Time 3D Reconstruction and Streaming for Immersive Multi-Client Live Telepresence

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

> Patrick Stotko, Stefan Krumpen, Matthias B. Hullin, Michael Weinmann, and Reinhard Klein.
> "SLAMCast: Large-Scale, Real-Time 3D Reconstruction and Streaming for Immersive Multi-Client Live Telepresence."
> *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, 2019.
> DOI: 10.1109/TVCG.2019.2899231

5.1 Summary of the Publication

This work addresses the problem of efficiently reconstructing and streaming a 3D scene between remotely connected clients for immersive rendering in Virtual Reality using head-mounted displays (HMDs). For this purpose, a scalable multi-client telepresence system has been developed where a single user captures their environment using a low-cost RGB-D camera which can then be independently explored as a 3D model by an arbitrary number of further users. Thus, instead of directly visualizing the recorded raw video data or 360° images which are restricted to the position of the capturing device and require fast and immediate streaming to ensure low latencies, we use reconstruction-based approaches to decouple these requirements. In particular, our key insight is that a high degree of immersion with low latencies can be achieved by real-time rendering of the captured 3D model whereas the incremental streaming of that model can only be performed at lower interactive framerates.

The system itself consists of three major components: 1) the reconstruction client, 2) the central server, and 3) an arbitrary number of exploration clients. On the capturing side, we leveraged established methods for volumetric 3D reconstruction [Nießner et al., 2013; Kähler et al., 2015] to build a large-scale 3D model incrementally from a sequence of RGB-D images using spatially-hashed voxel data structures. Here, we used the fact that visible voxel blocks that leave the view current frustum of the camera will no longer receive updates until they become visible again. Consequently, we added those blocks to a stream set, which is implemented as a GPU hash set data structure similar to its hash map counterpart, to allow controlling the amount of streamed data per frame. We also proposed a pre-fetching functionality to stream the currently visible scene parts, e.g. when the user stops moving the camera and points to a certain location, as well as a partial reset functionality to quickly update the currently visible scene parts and handle dynamics in quasi-static scenes. In particular, pre-fetching is only performed if the size of the stream set after filtering by an exponential moving average over a time period of $\tau = 5$ seconds is below a threshold. On the server side, we managed the streamed voxel block data using both the original voxel data structure, which is primarily designed for data fusion, and a bandwidth-optimized version for efficient streaming to the exploration clients. Surface information, i.e. the truncated signed distance function (TSDF) value (4 bytes) and its associated weight (4 bytes), are converted to a Marching Cubes [Lorensen and Cline, 1987] index (1 byte) which directly encodes the relevant triangulation information of a voxel to its neighbors but discards interpolation information leading to a lossy compression. Based on this index, color values at voxels, where no surface geometry will be generated, are cut off to improve the data compression. Similar to the handling of updates in the reconstruction client, the server maintained a stream set for each exploration client which allows each client to independently provide an advanced streaming strategy and, in turn, enables recovery from network outages. On the exploration side, this flexibility is used to request scene updates either in the order of reconstruction, the currently visible blocks based on the HMD's field of view and pose in world space, or in a random order. After receiving new data from the server, the mesh as well as the additional three coarser levels of detail, which are computed to accelerate the rendering of distant scene parts, are updated. In order to improve the collaboration experience in VR, the pose information of the reconstruction client and each connected exploration client is broadcast via the server such that all users can observe their movement in the virtual scene. Furthermore, the exploration client can request the current RGB image from the reconstruction client to display finer structures such as text that may be below the reconstruction resolution.

Besides the efficient storage of the 3D data, the fast and reliable management of this data is a crucial component of our telepresence system. In contrast to previous hash map data structures [Nießner et al., 2013; Kähler et al., 2015] which handle insertion or removal failures by considering subsequent input frames and, hence, relying on the high sensor framerate, our GPU data structures provided stronger guarantees to avoid data loss and to maintain a consistent state of the system. For this purpose, all concurrent operations maintained the hash entry positions and all links to colliding elements as an invariant.

We analyzed the performance of our system in terms of bandwidth requirements and visual scene quality. Throughout several real-world datasets, our Marching Cubes index-based data structure reduced the amount of required bandwidth by over 90 % in comparison to the TSDF-based voxel data structure. We also generally observed a similar visual quality of the compressed 3D scene model due to the good reconstruction accuracy at 5 mm voxel resolution. Small artifacts may be introduced in regions with highly textured objects or sharp edges which, however, could be compensated by requesting the current RGB image from the reconstruction client.

In summary, we introduced a scalable live telepresence systems that streams 3D scene information using a bandwidth-efficient voxel data structure and manages the stream states with a fast and reliable GPU hash map and hash set data structure.

5.2 Author Contributions of the Publication

In this work, I developed the design of the hash map and hash set data structure as well as the logic of the reconstruction client and the server components. The novel bandwidth-efficient voxel data structure for streaming data between the server and the exploration client has been proposed by both Stefan Krumpen and me. Furthermore, Stefan Krumpen developed and implemented the exploration client component. Finally, I performed the evaluation and experiments of the bandwidth and scalability of the system as well as of the performance and reliability of the hash data structures.

Chapter 6

Efficient 3D Reconstruction and Streaming for Group-Scale Multi-Client Live Telepresence

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

Patrick Stotko, Stefan Krumpen, Michael Weinmann, and Reinhard Klein.
"Efficient 3D Reconstruction and Streaming for Group-Scale Multi-Client Live Telepresence."
IEEE International Symposium on Mixed and Augmented Reality (ISMAR), 2019.
DOI: 10.1109/ISMAR.2019.00018

6.1 Summary of the Publication

While we proposed a live telepresence system [Stotko et al., 2019a] in the previous chapter that focused on providing immersive exploration experiences to an arbitrary number of users in a reconstructed quasi-static 3D virtual model, the focus of this work lies on the optimization of the reconstruction component to overcome the limitations of such systems in terms of client scalability and streaming latency.

Our optimizations targeted several stages of the volumetric 3D reconstruction pipeline: 1) the image preprocessing stage, 2) the data fusion stage, and 3) the model visualization stage. Furthermore, the third optimization can also be applied to the server component of the telepresence system to provide the same consistent model for both the local user at the capturing side as well as the remote users. At the preprocessing stage, we improved the robustness of the captured RGB-D image data by filtering out unreliable depth measurements. For this purpose, we detected samples at depth discontinuities in a small window similar to previous work [Whelan et al., 2015a] and, in addition, also detected samples in regions with a low relative number of valid samples which indicates that data acquisition is potentially unreliable in these regions and could result in significantly distorted samples caused by stronger noise or even systematic bias. During the data fusion stage, we directed our attention onto the allocation of the voxel blocks which is performed to only explicitly manage data within a small band around the surface [Nießner et al., 2013; Kähler et al., 2015]. Since noisy samples can unnecessarily increase this allocated region and might not always be filtered out by the previous optimization, we applied an implicit downsampling filter on the depth image to only consider a subset of the samples when determining the set of voxel blocks that should be allocated. In contrast to an additional garbage collection step, this further reduced the cost of the actual update step within the data fusion stage. Finally at the visualization stage of the standalone volumetric 3D reconstruction pipeline, we classified voxels with a low weight as unstable similar to previous work [Keller et al., 2013], but, instead of deleting these voxels, we only ignored them in the raycasting process as well as in the Marching Cubes [Lorensen and Cline, 1987] algorithm. This effectively filtered out noisy regions in the 3D model that may still become stable in the future when more captured data from the camera is fused. A similar optimization has been incorporated in the server component of the telepresence system. During the integration of the updated scene parts from the reconstruction client into the global model, we only allocated and updated the bandwidth-efficient voxel block data, consisting of a Marching Cubes index (1 byte) as well as a color value (3 bytes) per voxel, which actually contained stable surface information. Therefore, all voxel blocks that would not contribute to the visualization in VR are discarded and not queued to the stream sets of the connected exploration clients.

We evaluated the performance of each proposed optimization under the aspects of scalability and streaming latency of the telepresence system as well as visual quality of the reconstructed 3D model. For the analysis of the system scalability, we measured the maximum number of exploration clients that the server could handle in real time without introducing an increase in the overall delay. Whereas the base system [Stotko et al., 2019a] could only handle up to about five clients simultaneously, each optimization contributed towards a higher overall performance upon this baseline allowing the full system to handle groups of more than 24 clients. Furthermore, we observed a significantly lower streaming latency between the server and the exploration clients since the representation of the 3D model in terms of the set of spatially-hashed voxel blocks was more compact. These efficiency improvements can also be seen in standalone 3D reconstruction applications where we compared the visual quality against the baseline system. In addition to reduced memory requirements and runtime by up to 40 % and 60 % respectively, our proposed optimizations also reduced the amount of artifacts introduced by noisy input data which resulted in an overall higher visual quality. Although the completeness of the 3D model slightly decreased at the same time, this could provide an additional implicit guidance to the user for putting more attention in capturing these affected regions as they require more reliable data to obtain a smooth and consistent reconstruction.

In summary, we introduced several optimizations to improve the overall performance of our live telepresence system which opens up several new applications in education or collaboration scenarios that involve larger groups of people. We also demonstrated that these contributions are not limited to telepresence applications and could also be beneficial for standalone volumetric 3D reconstruction approaches in general.

6.2 Author Contributions of the Publication

In this work, I developed the proposed optimizations to the volumetric 3D reconstruction algorithm as well as to the streaming components of the telepresence system. Furthermore, I performed the quantitative and qualitative experiments and evaluations concerning scalability, latency, and visual quality.

Chapter 7

Albedo estimation for real-time 3D reconstruction using RGB-D and IR data

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

Patrick Stotko, Michael Weinmann, and Reinhard Klein.
"Albedo estimation for real-time 3D reconstruction using RGB-D and IR data."
ISPRS Journal of Photogrammetry and Remote Sensing (P&RS), 2019.
DOI: `10.1016/j.isprsjprs.2019.01.018`

7.1 Summary of the Publication

In the scope of this work, we considered the problem of jointly reconstructing geometry and appearance information of static scenes from RGB-D image data in real time. Whereas most RGB-D image-based reconstruction approaches solely concentrated on the fusion of the scene geometry and the captured RGB data, we estimated the surface appearance in terms of the spatially-varying albedo by exploiting the infrared (IR) data that current time-of-flight sensors such as the Microsoft Kinect v2 record to compute depth information.

Our system is built upon established volumetric 3D reconstruction approaches [Nießner et al., 2013; Kähler et al., 2015] for depth data fusion into a single model. Considering the IR images, we first leveraged the controlled illumination setup of the sensor to model the perceived radiance in the infrared domain by a direct illumination term, where the incoming light direction is approximated by the view direction, as well as a constant ambient term, which approximates the indirect illumination components [Or-El et al., 2016]. We iteratively optimized for both the ambient term and the pixel-wise IR albedo in terms of an energy formulation with Total Variation regularization where the L_1-based smoothness term is weighted by the magnitude of the normal image gradient $\nabla \boldsymbol{n}$ which served as a curvature-related metric and ensured sharp edges in the IR albedo image at object boundaries. Afterwards, we used the estimated IR albedo to guide the optimization of the RGB albedo from the ambiguous intrinsic image decomposition formulation by adding a coupling term [Kerl et al., 2014]. However, since the reflectance properties between different wavelengths could significantly vary between materials which, in particular, includes the visible spectrum as well as infrared light, a constant image-wide coupling term is not sufficient to model this behavior. In order to address this issue, we employed image segmentation to identify regions with similar reflectance properties and considered several approaches using either color-based or geometry-based clustering. The resulting probability images were used to formulate a segment-wise coupling term for the albedo image optimization. We further replaced the smoothness term of the involved shading image by a temporal dampening term to reduce the cost of the optimization process and to improve its convergence. In order to compute the solutions of the Total Variation-based energy formulations, we developed an approximate primal-dual solver where we modified the evaluation of the partial derivative of the data term to allow for an efficient computation in parallel on the GPU. Furthermore, we accelerated this process by projecting the solution from the previous frame into the current one using the estimated camera pose from the geometry reconstruction pipeline which then served as an initialization and, thereby, enabled distributing the computational cost across multiple input images of the sensor. During the data fusion stage, we further accounted for this acceleration by incorporating confidence weights based on the total number of TV solver iterations spent per pixel across various time steps.

We evaluated the performance of our joint geometry and albedo reconstruction approach in terms of runtime performance and convergence of the albedo image estimation as well as visual quality of the final reconstructed 3D model between the considered segmentation approaches. Our localized temporally-coherent shading term formulation resulted in a higher convergence rate, especially during the initial iterations, while at the same time the required total runtime of the estimation process was up to 20 % lower than the baseline Total Variation approach. Similarly, the improved initialization of the TV solver and the significantly reduced number of iterations per frame further reduced the runtime cost, but introduced artifacts in those parts of the albedo image which were not yet converged. The additional confidence-based weighting in the data fusion step eliminated these artifacts resulting in a similar quality of the final reconstructed surface albedo as the considerably slower baseline approach. We also analyzed the effect of different segmentation approaches to the quality of the estimated albedo images. A major challenge of the geometry-based and color-based hard clustering approaches was occurring over- and under-segmentation which resulted in wrongly classified object boundaries and, thereby, inconsistent albedo values within a single material. In contrast to this, the probability images of the color-based soft clustering were smooth across the materials and improved the overall accuracy of the reconstructed albedo.

In summary, we introduced a novel approach for reconstructing both geometry and appearance information in terms of the spatially-varying albedo by modeling the relation of the albedo across different wavelengths in a segment-wise manner and exploiting temporal coherence between input frames to accelerate the optimization of the Total Variation-based energy formulations in real time.

7.2 Author Contributions of the Publication

In this work, I developed the real-time albedo estimation and fusion algorithm and performed the experimental evaluation of the approach. Parts of this work have been already published in my master's thesis [Stotko, 2016a] which, in particular, covers the Total Variation-based optimization of the albedo inspired by Kerl et al. [2014] as well as the involved approximate Total Variation solver. In contrast to these components, I introduced several further novel contributions and extensions in the scope of this publication which go beyond the state of the master's thesis. This includes 1) the introduction of image segmentation based on the color and geometry information to improve the coupling between the RGB and IR images, 2) the localized shading estimation with temporal dampening replacing the costly Total Variation-based optimization of the shading term, 3) the acceleration of the TV solver using temporal information from the previous frame, and 4) the extension of the albedo fusion step using confidence weights from the TV solver.

Part III

Conclusion

Chapter 8

Conclusion

In this chapter, we provide a summary of the contributions of this thesis (see Section 8.1). Furthermore, we discuss the limitations of our work as well as potential future research directions (see Section 8.2).

8.1 Summary of Contributions and Impact

In the scope of this thesis, we presented three key contributions which, in particular, include a family of generalized LOP operators for data prefiltering, a scalable 3D telepresence system based on large-scale surface reconstruction, as well as a segment-wise albedo estimation method from RGB-D and IR data.

Generalized LOP Operators for Filtering. In the context of data prefiltering, we directed our attention onto the LOP operator which does not require structured input data in terms of 2D images like in the case of bilateral filters, but instead allows to filter unstructured 3D point clouds. In Chapter 4, we studied the operator in more detail and revealed that the involved attraction and repulsion terms are both closely related to the Mean Shift framework. Furthermore, we derived the corresponding kernel in the context of Mean Shift and generalized this result from the initial scenario of L_1-based estimators to arbitrary localized L_p estimators which culminated in the definition of a novel family of kernels. In order to obtain more insights on the general structure and behavior of the LOP operator, we also derived various theoretical properties of the kernel family and illustrated their application in a variety of related scenarios. We demonstrated that our proposed density weighting scheme as well as our estimate of the kernel approximation required for the closed-form Continuous LOP (CLOP) operator consistently outperforms previous established results. Finally, we introduced a set of robust loss functions, which correspond to the respective kernels, and showed that their application in the closely related field of mesh denoising for filtering surface normals in a local manner leads to more detailed results in comparison to the corresponding global variants.

Scalable 3D Telepresence based on Surface Reconstruction. We further investigated techniques to incorporate real-time surface reconstruction techniques into 3D telepresence systems to improve their practicality and to extend their scope to further application scenarios. To this end, we presented a multi-client telepresence system in Chapter 5 for sharing live-reconstructed 3D scene models in real time to an arbitrary number of remote users who can independently explore and interact with the model in VR. We proposed an efficient volumetric data structure based on Marching Cubes indices for streaming data between the server component and the exploration clients and showed that the bandwidth requirements of the system can be drastically reduced with only slight reductions in the reconstruction accuracy. Furthermore, we developed a reliable GPU hash map and set data structure which is employed at the reconstruction client as well as on the server to efficiently manage the individual states of the streamed scene data of all connected clients. This allowed the system to recover from network outages of individual clients and further increased its practicality in real-world scenarios.

In Chapter 6, we proposed several algorithmic improvements for the reconstruction client as well as for the server component to further increase the compactness of the streamed 3D model. We extended the image preprocessing step of the reconstruction client by detecting unreliable depth samples in the captured frames. In addition to approaches from previous work which determine whether a sample lies at a depth discontinuity, we further marked depth values as unreliable if their local neighborhood only contains a low number of other valid samples which provides an indication that data acquisition in such regions can be more challenging. Furthermore, we reduced the likelihood of unnecessary enlargements of the allocated band of voxel blocks by considering only a subset of the depth samples in the allocation step which corresponds to an implicit downsampling filter applied to the depth image. Finally, we restricted the computation of our bandwidth-efficient voxel data structure to stable fusion results and demonstrated that with the incorporation of all developed extensions the scalability of our 3D telepresence system improved significantly.

Our work also received significant attention from the open-source community and from industry. We released the implementation of our GPU hash map and set data structures along with a thorough documentation [Stotko, 2019b] which has been recently integrated into the popular Open3D library [Zhou et al., 2018]. Furthermore, the software of our telepresence system has been licensed by the company DoubleMe [University of Bonn, 2021].

Segment-wise IR-Guided Albedo Estimation for 3D Reconstruction. In the context of appearance reconstruction, we integrated albedo estimation into common real-time 3D reconstruction systems and analyzed the effect of captured infrared data from current time-of-flight cameras as an additional guidance in more detail. To this end, we developed several contributions in the scope of this thesis which have been presented in Chapter 7 and go beyond our previous work [Stotko, 2016a]. In particular, we studied image segmentation approaches which either take color or geometry information into consideration to identify clusters of objects with similar reflectance properties between the visible and the infrared spectrum. Furthermore, we proposed a temporally-damped local shading prior term as well as an improved initialization of the involved Total Variation solver using temporal information. In conjunction with an additional confidence-based weighting scheme applied in the albedo fusion step, we showed that our contributions improved the performance and robustness of the IR-guided intrinsic image decomposition process.

8.2 Limitations and Future Work

While some challenges in real-time 3D reconstruction have been addressed by the methods developed in this thesis, more work is required to address the remaining limitations. To this end, we provide an outlook of potential future research directions as well as further open challenges in this context.

8.2.1 Data Prefiltering

Surface Projection for Registration. Recently, the projection of points onto a continuous surface defined by Hermite radial basis function implicits has been used as a replacement of the surface evaluation step of real-time surfel-based reconstruction systems which is typically implemented in terms of standard surfel splatting [Xu et al., 2022b]. In combination with further curvature and confidence-based weighting schemes, the robustness of the registration process in the camera tracking stage to noisy input data has been improved. A potential future direction could include investigations towards enhancing real-time registration with other projection-based operators such as the LOP operator, which was specifically designed for the presence of strong noise, or our generalization in terms of L_p estimators, which has been proposed in Chapter 4. In this context, a tighter integration of the operator into the optimization process could be analyzed.

Deblurring of Color Images. The Gaussian loss function, which is also often referred to as the Welsch loss, has been recently studied for image deblurring to improve the robustness in the scenarios with outliers and saturated regions [Xu et al., 2022c]. Further investigations in this direction could explore the benefits of applying our family of robust loss function that correspond to the generalized LOP operators. Therefore, incorporating such deblurring methods in the reconstruction process could not only lead to sharper and more detailed color textures in the final reconstructed model, but may also contribute towards enhancing the accuracy of the color-based components in camera tracking [Liu et al., 2021a].

Priors for Learning-based Surface Reconstruction. Ideas from classical surface reconstruction techniques were recently incorporated as an additional guidance for learning signed distance functions in a fully supervised [Liu et al., 2021b] or self-supervised way [Wang et al., 2021]. In a more general scenario which is not necessarily limited to implicit data representations, the density-based projection behavior of LOP operators could be employed as a prior to steer the learning process in a more controlled way and, in turn, potentially improve the convergence rate.

8.2.2 3D Telepresence

Reconstruction of Articulated Objects. Although quasi-static scenes can in principle also be reconstructed and streamed in our telepresence system which has been introduced in Chapters 5 and 6, non-static scene parts and objects were handled in an oversimplified manner by erasing the corresponding parts from the global 3D model based on the current view frustum. Afterwards, the object was recaptured again to obtain an updated version of the quasi-static scene. In future work, approaches for detecting articulated objects as well as their kinematics in terms of the involved joint parameters [Li et al., 2020a] could be investigated to obtain a more complete virtual 3D model. This way, the initially hidden interior of furniture such as cabinets could be reconstructed and remote users would be able to virtually manipulate the semantically enriched scene which enables further collaborative interaction scenarios.

Streaming and Visualizing Globally Consistent Models. In addition to the sparse nature of the voxel block data structure, the reconstruction client of our telepresence system further relied on a frame-to-model-based camera tracking approach to only update the 3D scene model in a local manner and, in turn, to reduce the amount of updated data for streaming. Since camera drift cannot be corrected in this way during reconstruction, globally consistent tracking methods should be employed instead. In this context, we considered continuous reintegration of the input data [Dai et al., 2017] with the updated camera poses which, however, drastically increased the bandwidth requirements as the virtual model was globally deformed. Alternative approaches maintain a set of static submaps and only update their poses, but postpone the final fusion into a single model after the end of the capturing session or use simple depth-based foreground rendering for overlapping parts which may lead to artifacts [Kähler et al., 2016a; Golodetz et al., 2018]. Therefore, further exploring globally consistent 3D reconstruction in the context of practical VR applications to meet the higher demands in terms of visualization quality and streaming efficiency would be an interesting line of future research.

Streaming Neural Scene Representations. Recent learning-based methods focused on continuous surface representations with neural networks to avoid the inherent limitation of discretized data structures. While respective online approaches are still limited in terms of reconstruction resolution when processing a small set of keyframes in real time [Yan et al., 2021; Ortiz et al., 2022], an interesting future direction would be their incorporation into reconstruction-based telepresence systems. In particular, exploring ways how such progressively updated neural representations can be efficiently streamed to other remote users in a space and runtime-efficient manner would not only be beneficial in the field of telepresence systems research, but could also contribute towards a deeper understanding of the evolution of such neural data models over time.

Guidance in Teleoperation Applications. Based on our 3D telepresence system, we also explored its application to teleoperation scenarios [Stotko et al., 2019c] in subsequent work. We showed that employing live-captured 3D scene models and allowing users to independently explore the scene in VR resulted in a higher situation awareness over video-based solutions and allowed the users to remotely navigate a ground robot [Schwarz et al., 2019] more precisely through a challenging environment with many obstacles. In a similar scenario, Zhang et al. [2021] recently considered scene capturing with an unmanned aerial vehicle (UAV) which is remotely controlled by a user in VR from the first-person perspective of the drone. Further investigations in this direction could include the scanning of large environments with possibly multiple autonomous robots by only providing sparse guidance and hints from VR remote operators in captured regions which require closer attention.

8.2.3 Appearance Reconstruction

Segment-wise Temporal Acceleration. The techniques for accelerating the reconstruction of albedo data using forward-projected temporal information of estimates from previous time frames, which have been presented in Chapter 7, enabled the overall reconstruction system to run in real time. However, convergence artifacts may occur in regions where no depth data is available to properly compute the corresponding pixel coordinates for forward projection or where strong variations in the shading image from high-frequency illumination lead to suboptimal initializations. In future work, more sophisticated propagation strategies could be investigated which may for instance reuse the already computed segmentation information for the current image to identify similar regions and perform boundary-preserving hole-filling and correction of the data.

IR-Guided Neural Reflectance Estimation. Similar to the recent developments in learning-based geometry reconstruction, methods based on neural networks were also used to infer a representation of scene colors or even scene appearance information in terms of diffuse albedo and specular model parameters [Bi et al., 2020]. Although these approaches are still limited to small object-scale scenes and are optimized in an offline manner using full supervision, an interesting future direction could include investigations towards incorporating the IR data from RGB-D cameras as an additional source of information. This, in turn, could provide further guidance in large room-scale scenarios where objects are typically reconstructed from a set of less uniformly distributed camera angles.

Bibliography

Alain, Guillaume and Yoshua Bengio (2014). “What Regularized Auto-Encoders Learn from the Data-Generating Distribution.” *Journal of Machine Learning Research*.

Alcantara, Dan A., Andrei Sharf, Fatemeh Abbasinejad, Shubhabrata Sengupta, Michael Mitzenmacher, John D. Owens, and Nina Amenta (2009). “Real-time Parallel Hashing on the GPU.” *ACM Transactions on Graphics (TOG)*.

Alcantara, Dan Anthony Feliciano (2011). “Efficient Hash Tables on the GPU.” *Ph.D. thesis. University of California, Davis*.

Alexa, Marc, Johannes Behr, Daniel Cohen-Or, Shachar Fleishman, David Levin, and Claudio T. Silva (2003). “Computing and Rendering Point Set Surfaces.” *IEEE Transactions on Visualization and Computer Graphics (TVCG)*.

Amanatides, John and Andrew Woo (1987). “A Fast Voxel Traversal Algorithm for Ray Tracing.” *Annual Conference of the European Association for Computer Graphics (Eurographics)*.

Amenta, Nina and Yong Joo Kil (2004). “Defining Point-Set Surfaces.” *ACM Transactions on Graphics (TOG)*.

Anderson, Ben (2004). *An Implementation of the Marching Cubes Algorithm*. `https://www.cs.carleton.edu/cs_comps/0405/shape/marching_cubes.html`. Accessed: 2022-09-21.

Ashkiani, Saman, Martin Farach-Colton, and John D. Owens (2018). “A Dynamic Hash Table for the GPU.” *IEEE International Parallel and Distributed Processing Symposium*.

Avron, Haim, Andrei Sharf, Chen Greif, and Daniel Cohen-Or (2010). “ℓ_1-Sparse Reconstruction of Sharp Point Set Surfaces.” *ACM Transactions on Graphics (TOG)*.

Azinović, 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)*.

Azinović, Dejan, Ricardo Martin-Brualla, Dan B. Goldman, Matthias Nießner, and Justus Thies (2022). "Neural RGB-D Surface Reconstruction." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Bai, Huidong, Prasanth Sasikumar, Jing Yang, and Mark Billinghurst (2020). "A User Study on Mixed Reality Remote Collaboration with Eye Gaze and Hand Gesture Sharing." *ACM Conference on Human Factors in Computing Systems (CHI)*.

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 (2015). "Shape, Illumination, and Reflectance from Shading." *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*.

Barrow, Harry G. and J. Martin Tenenbaum (1978). "Recovering intrinsic scene characteristics." *Computer Vision Systems*.

Bateman, Harry (1953). *Higher Transcendental Functions*.

Beck, Gaël, Tarn Duong, Mustapha Lebbah, Hanane Azzag, and Christophe Cérin (2019). "A Distributed and Approximated Nearest Neighbors Algorithm for an Efficient Large Scale Mean Shift Clustering." *Journal of Parallel and Distributed Computing*.

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

Bi, Sai, Zexiang Xu, Pratul Srinivasan, Ben Mildenhall, Kalyan Sunkavalli, Miloš Hašan, Yannick Hold-Geoffroy, David Kriegman, and Ravi Ramamoorthi (2020). "Neural Reflectance Fields for Appearance Acquisition." *arXiv preprint arXiv:2008.03824*.

Bigdeli, Siavash A. and Matthias Zwicker (2018). "Image Restoration using Autoencoding Priors." *International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*.

Bigdeli, Siavash A., Matthias Zwicker, Paolo Favaro, and Meiguang Jin (2017). "Deep Mean-Shift Priors for Image Restoration." *Advanced Neural Information Processing Systems (NeurIPS)*.

Black, Michael J., Guillermo Sapiro, David H. Marimont, and David Heeger (1998). "Robust Anisotropic Diffusion." *IEEE Transactions on Image Processing*.

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`.

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

Botelho, Fabiano C., Rasmus Pagh, and Nivio Ziviani (2013). "Practical Perfect Hashing in Nearly Optimal Space." *Information Systems*.

Calakli, Fatih and Gabriel Taubin (2011). "SSD: Smooth Signed Distance Surface Reconstruction." *Computer Graphics Forum (CGF)*.

Cao, Yan-Pei, Leif Kobbelt, and Shi-Min Hu (2018). "Real-time High-accuracy Three-Dimensional Reconstruction with Consumer RGB-D Cameras." *ACM Transactions on Graphics (TOG)*.

Carr, Jonathan C., Richard K. Beatson, Jon B. Cherrie, Tim J. Mitchell, W. Richard Fright, Bruce C. McCallum, and Tim R. Evans (2001). "Reconstruction and Representation of 3D Objects with Radial Basis Functions." *Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*.

Casas, Dan, Andrew Feng, Oleg Alexander, Graham Fyffe, Paul Debevec, Ryosuke Ichikari, Hao Li, Kyle Olszewski, Evan Suma, and Ari Shapiro (2016). "Rapid Photorealistic Blendshape Modeling from RGB-D Sensors." *Conference on Computer Animation and Social Agents Computer Animation and Virtual Worlds*.

Chakareski, Jacob (2017). "VR/AR Immersive Communication: Caching, Edge Computing, and Transmission Trade-Offs." *Workshop on Virtual Reality and Augmented Reality Network*.

Chambolle, Antonin and Thomas Pock (2011). "A First-Order Primal-Dual Algorithm for Convex Problems with Applications to Imaging." *Journal of Mathematical Imaging and Vision*.

Chen, Haolan, Bi'an Du, Shitong Luo, and Wei Hu (2021a). "Deep Point Set Resampling via Gradient Fields." *arXiv preprint arXiv:2111.02045*.

Chen, Honghua, Mingqiang Wei, Yangxing Sun, Xingyu Xie, and Jun Wang (2019). "Multi-Patch Collaborative Point Cloud Denoising via Low-Rank Recovery with Graph Constraint." *IEEE Transactions on Visualization and Computer Graphics (TVCG)*.

Chen, Honghua, Zeyong Wei, Xianzhi Li, Yabin Xu, Mingqiang Wei, and Jun Wang (2022a). "RePCD-Net: Feature-Aware Recurrent Point Cloud Denoising Network." *International Journal of Computer Vision (IJCV)*.

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

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

Chen, Ting-Li (2015). "On the convergence and consistency of the blurring mean-shift process." *Annals of the Institute of Statistical Mathematics*.

Chen, Ye, Jinxian Liu, Bingbing Ni, Hang Wang, Jiancheng Yang, Ning Liu, Teng Li, and Qi Tian (2021b). "Shape Self-Correction for Unsupervised Point Cloud Understanding." *IEEE International Conference on Computer Vision (ICCV)*.

Chen, Zhiqin, Thomas Funkhouser, Peter Hedman, and Andrea Tagliasacchi (2022b). "MobileNeRF: Exploiting the Polygon Rasterization Pipeline for Efficient Neural Field Rendering on Mobile Architectures." *arXiv preprint arXiv:2208.00277*.

Cheng, Xuan, Ming Zeng, Jinpeng Lin, Zizhao Wu, and Xinguo Liu (2019a). "Efficient L_0 resampling of point sets." *Computer Aided Geometric Design*.

Cheng, Yizong (1995). "Mean Shift, Mode Seeking, and Clustering." *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*.

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)*.

Choe, Gyeongmin, Jaesik Park, Yu-Wing Tai, and In So Kweon (2017). "Refining Geometry from Depth Sensors using IR Shading Images." *International Journal of Computer Vision (IJCV)*.

Choe, Gyeongmin, Jaesik Park, Yu-Wing Tai, and In So Kweon (2014). "Exploiting Shading Cues in Kinect IR Images for Geometry Refinement." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Choe, Jaesung, Byeongin Joung, Francois Rameau, Jaesik Park, and In So Kweon (2022). "Deep Point Cloud Reconstruction." *International Conference on Learning Representations (ICLR)*.

Collet, Y. and C. Turner (2016). *Smaller and faster data compression with Zstandard*. `https://code.fb.com/core-data/smaller-and-faster-data-compression-with-zstandard/`. Accessed: 2019-01-29.

Comaniciu, Dorin and Peter Meer (2002). "Mean Shift: A Robust Approach toward Feature Space Analysis." *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*.

Corbillon, Xavier, Gwendal Simon, Alisa Devlic, and Jacob Chakareski (2017). "Viewport-adaptive navigable 360-degree video delivery." *IEEE International Conference on Communications*.

Curless, Brian and Marc Levoy (1996). "A Volumetric Method for Building Complex Models from Range Images." *Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*.

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." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Dervieux, Alain and François Thomasset (1979). "A finite element method for the simulation of a Rayleigh-Taylor instability." *Approximation Methods for Navier-Stokes Problems*.

Digne, Julie, Sébastien Valette, and Raphaëlle Chaine (2017). "Sparse Geometric Representation Through Local Shape Probing." *IEEE Transactions on Visualization and Computer Graphics (TVCG)*.

Dinesh, Chinthaka, Gene Cheung, and Ivan V. Bajić (2020). "Point Cloud Denoising via Feature Graph Laplacian Regularization." *IEEE Transactions on Image Processing*.

Dong, Siyan, Kai Xu, Qiang Zhou, Andrea Tagliasacchi, Shiqing Xin, Matthias Nießner, and Baoquan Chen (2019). "Multi-Robot Collaborative Dense Scene Reconstruction." *ACM Transactions on Graphics (TOG)*.

Dong, Wei, Jieqi Shi, Weijie Tang, Xin Wang, and Hongbin Zha (2018). "An Efficient Volumetric Mesh Representation for Real-time Scene Reconstruction using Spatial Hashing." *IEEE International Conference on Robotics and Automation (ICRA)*.

Dou, Mingsong and Henry Fuchs (2014). "Temporally Enhanced 3D Capture of Room-sized Dynamic Scenes with Commodity Depth Cameras." *IEEE Virtual Reality Conference*.

Dou, Mingsong, Sameh Khamis, Yury Degtyarev, Philip Davidson, Sean Ryan Fanello, Adarsh Kowdle, Sergio Orts-Escolano, Christoph Rhemann, David Kim, Jonathan Taylor, Pushmeet Kohli, Vladimir Tankovich, and Shahram Izadi (2016). "Fusion4D: Real-time Performance Capture of Challenging Scenes." *ACM Transactions on Graphics (TOG)*.

Duan, Chaojing, Siheng Chen, and Jelena Kovacevic (2019). "3D Point Cloud Denoising via Deep Neural Network based Local Surface Estimation." *IEEE International Conference on Acoustics, Speech, and Signal Processing*.

Or-El, Roy, Rom Hershkovitz, Aaron Wetzler, Guy Rosman, Alfred M. Bruckstein, and Ron Kimmel (2016). "Real-time Depth Refinement for Specular Objects." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Fairchild, Allen J., Simon P. Campion, Arturo S. García, Robin Wolff, Terrence Fernando, and David J. Roberts (2016). "A Mixed Reality Telepresence System for Collaborative Space Operation." *IEEE Transactions on Circuits and Systems for Video Technology*.

Fan, Ching-Ling, Jean Lee, Wen-Chih Lo, Chun-Ying Huang, Kuan-Ta Chen, and Cheng-Hsin Hsu (2017). "Fixation Prediction for 360° Video Streaming in Head-Mounted Virtual Reality." *Workshop on Network and Operating Systems Support for Digital Audio and Video*.

Fankhauser, Péter, Michael Bloesch, Diego Rodriguez, Ralf Kaestner, Marco Hutter, and Roland Siegwart (2015). "Kinect v2 for Mobile Robot Navigation: Evaluation and Modeling." *International Conference on Advanced Robotics (ICAR)*.

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

Fontaine, Gary (1992). "The Experience of a Sense of Presence in Intercultural and Int. Encounters." *Presence: Teleoperators and Virtual Environments*.

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)*.

Fuchs, Henry, Gary Bishop, Kevin Arthur, Leonard McMillan, Ruzena Bajcsy, Sang Lee, Hany Farid, and Takeo Kanade (1994). "Virtual Space Teleconferencing Using a Sea of Cameras." *International Conference on Medical Robotics and Computer Assisted Surgery*.

Fuchs, Henry, Andrei State, and Jean-Charles Bazin (2014). "Immersive 3D Telepresence." *Computer*.

Fuhrmann, Simon and Michael Goesele (2011). "Fusion of Depth Maps with Multiple Scales." *ACM Transactions on Graphics (TOG)*.

Fuhrmann, Simon and Michael Goesele (2014). "Floating Scale Surface Reconstruction." *ACM Transactions on Graphics (TOG)*.

Fukunaga, Keinosuke and Larry Hostetler (1975). "The estimation of the gradient of a density function, with applications in pattern recognition." *IEEE Transactions on Information Theory*.

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)*.

García, Ismael, Sylvain Lefebvre, Samuel Hornus, and Anass Lasram (2011). "Coherent Parallel Hashing." *ACM Transactions on Graphics (TOG)*.

Ghassabeh, Youness Aliyari (2015). "A sufficient condition for the convergence of the mean shift algorithm with Gaussian kernel." *Journal of Multivariate Analysis*.

Golodetz, Stuart, Tommaso Cavallari, Nicholas A. Lord, Victor A. Prisacariu, David W. Murray, and Philip H. S. Torr (2018). "Collaborative Large-Scale Dense 3D Reconstruction with Online Inter-Agent Pose Optimisation." *IEEE Transactions on Visualization and Computer Graphics (TVCG)*.

Grillenzoni, Carlo (2016). "Design of Blurring Mean-Shift Algorithms for Data Classification." *Journal of Classification*.

Groueix, Thibault, Matthew Fisher, Vladimir G. Kim, Bryan C. Russell, and Mathieu Aubry (2018). "A Papier-Mâché Approach to Learning 3D Surface Generation." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Guarnera, Darya, Giuseppe Claudio Guarnera, Abhijeet Ghosh, Cornelia Denk, and Mashhuda Glencross (2016). "BRDF Representation and Acquisition." *Computer Graphics Forum (CGF)*.

Guo, Kaiwen, Feng Xu, Tao Yu, Xiaoyang Liu, Qionghai Dai, and Yebin Liu (2017). "Real-Time Geometry, Albedo, and Motion Reconstruction Using a Single RGB-D Camera." *ACM Transactions on Graphics (TOG)*.

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

Han, Lei and Lu Fang (2018). "FlashFusion: Real-time Globally Consistent Dense 3D Reconstruction using CPU Computing." *Robotics: Science and Systems*.

Handa, Ankur, Thomas Whelan, John McDonald, and Andrew J. Davison (2014). "A Benchmark for RGB-D Visual Odometry, 3D Reconstruction and SLAM." *IEEE International Conference on Robotics and Automation (ICRA)*.

Hanocka, Rana, Gal Metzer, Raja Giryes, and Daniel Cohen-Or (2020). "Point2Mesh: A Self-Prior for Deformable Meshes." *ACM Transactions on Graphics (TOG)*.

Held, Richard M. and Nathaniel I. Durlach (1992). "Telepresence." *Presence: Teleoperators and Virtual Environments*.

Henry, Peter, Dieter Fox, Achintya Bhowmik, and Rajiv Mongia (2013). "Patch Volumes: Segmentation-Based Consistent Mapping with RGB-D Cameras." *International Conference on 3D Vision (3DV)*.

Hermosilla, Pedro, Tobias Ritschel, and Timo Ropinski (2019). "Total Denoising: Unsupervised Learning of 3D Point Cloud Cleaning." *IEEE International Conference on Computer Vision (ICCV)*.

Hoppe, Hugues, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle (1992). "Surface Reconstruction from Unorganized Points." *Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*.

Horaud, Radu, Miles Hansard, Georgios Evangelidis, and Clément Ménier (2016). "An Overview of Depth Cameras and Range Scanners Based on Time-of-Flight Technologies." *Machine Vision and Applications*.

Hosseini, Mohammad and Viswanathan Swaminathan (2016). "Adaptive 360 VR Video Streaming: Divide and Conquer!" *IEEE International Symposium on Multimedia*.

Hou, Fei, Chiyu Wang, Wencheng Wang, Hong Qin, Chen Qian, and Ying He (2022). "Iterative Poisson Surface Reconstruction (iPSR) for Unoriented Points." *ACM Transactions on Graphics (TOG)*.

Hu, Liwen, Shunsuke Saito, Lingyu Wei, Koki Nagano, Jaewoo Seo, Jens Fursund, Iman Sadeghi, Carrie Sun, Yen-Chun Chen, and Hao Li (2017). "Avatar Digitization from a Single Image for Real-time Rendering." *ACM Transactions on Graphics (TOG)*.

Hu, Wei, Xiang Gao, Gene Cheung, and Zongming Guo (2020). "Feature Graph Learning for 3D Point Cloud Denoising." *IEEE Transactions on Image Processing*.

Hu, Wei, Qianjiang Hu, Zehua Wang, and Xiang Gao (2021). "Dynamic Point Cloud Denoising via Manifold-to-Manifold Distance." *IEEE Transactions on Image Processing*.

Huang, Chao, Ruihui Li, Xianzhi Li, and Chi-Wing Fu (2020a). "Non-Local Part-Aware Point Cloud Denoising." *arXiv preprint arXiv:2003.06631*.

Huang, Hui, Dan Li, Hao Zhang, Uri Ascher, and Daniel Cohen-Or (2009). "Consolidation of Unorganized Point Clouds for Surface Reconstruction." *ACM Transactions on Graphics (TOG)*.

Huang, Hui, Shihao Wu, Minglun Gong, Daniel Cohen-Or, Uri Ascher, and Hao Zhang (2013). "Edge-Aware Point Set Resampling." *ACM Transactions on Graphics (TOG)*.

Huang, Jiahui, Shi-Sheng Huang, Haoxuan Song, and Shi-Min Hu (2021). "DI-Fusion: Online Implicit 3D Reconstruction with Deep Priors." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

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

Huang, Kejun, Xiao Fu, and Nicholas Sidiropoulos (2018). "On Convergence of Epanechnikov Mean Shift." *AAAI Conference on Artificial Intelligence*.

Irfan, Muhammad Abeer and Enrico Magli (2021). "Joint Geometry and Color Point Cloud Denoising Based on Graph Wavelets." *IEEE Access*.

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

Jang, Jennifer and Heinrich Jiang (2021). "MeanShift++: Extremely Fast Mode-Seeking With Applications to Segmentation and Object Tracking." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Jin, Xudong and Yanfeng Gu (2017). "Superpixel-Based Intrinsic Image Decomposition of Hyperspectral Images." *IEEE Transactions on Geoscience and Remote Sensing*.

Jones, Brett, Rajinder Sodhi, Michael Murdock, Ravish Mehra, Hrvoje Benko, Andrew Wilson, Eyal Ofek, Blair MacIntyre, Nikunj Raghuvanshi, and Lior Shapira (2014). "RoomAlive: Magical Experiences Enabled by Scalable, Adaptive Projector-camera Units." *ACM Symposium on User Interface Software and Technology (UIST)*.

Kähler, Olaf, Victor A. Prisacariu, and David W. Murray (2016a). "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 (2015). "Very High Frame Rate Volumetric Integration of Depth Images on Mobile Devices." *IEEE Transactions on Visualization and Computer Graphics (TVCG)*.

Kähler, Olaf, Victor Prisacariu, Julien Valentin, and David Murray (2016b). "Hierarchical Voxel Block Hashing for Efficient Integration of Depth Images." *IEEE Robotics and Automation Letters*.

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

Kanade, Takeo, Peter Rander, and P. J. Narayanan (1997). "Virtualized reality: constructing virtual worlds from real scenes." *IEEE MultiMedia*.

Kazhdan, Michael, Matthew Bolitho, and Hugues Hoppe (2006). “Poisson Surface Reconstruction.” *Eurographics Symposium on Geometry Processing (SGP)*.

Kazhdan, Michael and Hugues Hoppe (2013). “Screened Poisson Surface Reconstruction.” *ACM Transactions on Graphics (TOG)*.

Kazhdan, Misha, Ming Chuang, Szymon Rusinkiewicz, and Hugues Hoppe (2020). “Poisson Surface Reconstruction with Envelope Constraints.” *Computer Graphics Forum (CGF)*.

Keller, Maik, Damien Lefloch, Martin Lambers, Shahram Izadi, Tim Weyrich, and Andreas Kolb (2013). “Real-Time 3D Reconstruction in Dynamic Scenes Using Point-Based Fusion.” *International Conference on 3D Vision (3DV)*.

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)*.

Khorasani, Farzad, Mehmet E. Belviranli, Rajiv Gupta, and Laxmi N. Bhuyan (2015). “Stadium Hashing: Scalable and Flexible Hashing on GPUs.” *International Conference on Parallel Architecture and Compilation*.

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)*.

Klingensmith, Matthew, Ivan Dryanovski, Siddhartha S. Srinivasa, and Jizhong Xiao (2015). “Chisel: Real Time Large Scale 3D Reconstruction Onboard a Mobile Device using Spatially Hashed Signed Distance Fields.” *Robotics: Science and Systems*.

Kolluri, Ravikrishna (2005). “Provably Good Moving Least Squares.” *ACM-SIAM Symposium on Discrete Algorithms*.

Kunert, André, Alexander Kulik, Stephan Beck, and Bernd Froehlich (2014). “Photoportals: Shared References in Space and Time.” *ACM Conference on Computer Supported Cooperative Work & Social Computing*.

Kurillo, Gregorij, Ruzena Bajcsy, Klara Nahrsted, and Oliver Kreylos (2008). “Immersive 3D Environment for Remote Collaboration and Training of Physical Activities.” *IEEE Virtual Reality Conference*.

Lambert, Johann Heinrich (1760). *Photometria sive de mensura et gradibus luminis, colorum et umbrae*.

Land, Edwin H. and John J. McCann (1971). “Lightness and Retinex Theory.” *Journal of the Optical Society of America*.

Lawrence, Jason, Dan B. Goldman, Supreeth Achar, Gregory Major Blascovich, Joseph G. Desloge, Tommy Fortes, Eric M. Gomez, Sascha Häberling, Hugues Hoppe, Andy Huibers, Claude Knaus, Brian Kuschak, Ricardo Martin-Brualla, Harris Nover, Andrew Ian Russel, Steven M. Seitz, and Kevin Tong (2021). "Project Starline: A high-fidelity telepresence system." *ACM Transactions on Graphics (TOG)*.

Leal, Esmeide, German Sanchez-Torres, and John W. Branch (2020). "Sparse Regularization-Based Approach for Point Cloud Denoising and Sharp Features Enhancement." *Sensors*.

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)*.

Lefebvre, Sylvain and Hugues Hoppe (2006). "Perfect Spatial Hashing." *ACM Transactions on Graphics (TOG)*.

Lefloch, Damien, Markus Kluge, Hamed Sarbolandi, Tim Weyrich, and Andreas Kolb (2017). "Comprehensive Use of Curvature For Robust And Accurate Online Surface Reconstruction." *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*.

Lefloch, Damien, Tim Weyrich, and Andreas Kolb (2015). "Anisotropic Point-Based Fusion." *International Conference on Information Fusion*.

Levin, David (2004). "Mesh-Independent Surface Interpolation." *Geometric Modeling for Scientific Visualization*.

Levoy, Marc, Kari Pulli, Brian Curless, Szymon Rusinkiewicz, David Koller, Lucas Pereira, Matt Ginzton, Sean Anderson, James Davis, Jeremy Ginsberg, Jonathan Shade, and Duane Fulk (2000). "The Digital Michelangelo Project: 3D Scanning of Large Statues." *Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*.

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, Tianye, Mira Slavcheva, Michael Zollhoefer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, Richard Newcombe, and Zhaoyang Lv (2022). "Neural 3D Video Synthesis from Multi-view Video." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Li, Xiangru, Zhanyi Hu, and Fuchao Wu (2007). "A note on the convergence of the mean shift." *Pattern Recognition*.

Li, Xiaolong, He Wang, Li Yi, Leonidas J. Guibas, A. Lynn Abbott, and Shuran Song (2020a). “Category-Level Articulated Object Pose Estimation.” *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Li, You and Javier Ibanez-Guzman (2020). “Lidar for Autonomous Driving: The Principles, Challenges, and Trends for Automotive Lidar and Perception Systems.” *IEEE Signal Processing Magazine*.

Li, Zhengqin, Mohammad Shafiei, Ravi Ramamoorthi, Kalyan Sunkavalli, and Manmohan Chandraker (2020b). “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)*.

Liao, Bin, Chunxia Xiao, Liqiang Jin, and Hongbo Fu (2013). “Efficient feature-preserving local projection operator for geometry reconstruction.” *Computer Aided Design*.

Lionar, Stefan, Lukas Schmid, Cesar Cadena, Roland Siegwart, and Andrei Cramariuc (2021). “NeuralBlox: Real-Time Neural Representation Fusion for Robust Volumetric Mapping.” *International Conference on 3D Vision (3DV)*.

Lipman, Yaron, Daniel Cohen-Or, David Levin, and Hillel Tal-Ezer (2007). “Parameterization-free Projection for Geometry Reconstruction.” *ACM Transactions on Graphics (TOG)*.

Liu, Peidong, Xingxing Zuo, Viktor Larsson, and Marc Pollefeys (2021a). “MBA-VO: Motion Blur Aware Visual Odometry.” *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Liu, Shengjun, Charlie CL Wang, Guido Brunnett, and Jun Wang (2016). “A Closed-Form Formulation of HRBF-Based Surface Reconstruction.” *Computer Aided Design*.

Liu, Shi-Lin, Hao-Xiang Guo, Hao Pan, Peng-Shuai Wang, Xin Tong, and Yang Liu (2021b). “Deep Implicit Moving Least-Squares Functions for 3D Reconstruction.” *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Liu, Yangdong, Wei Gao, and Zhanyi Hu (2018). “Geometrically stable tracking for depth images based 3D reconstruction on mobile devices.” *ISPRS Journal of Photogrammetry and Remote Sensing (P&RS)*.

Loop, Charles, Cha Zhang, and Zhengyou Zhang (2013). “Real-time High-resolution Sparse Voxelization with Application to Image-based Modeling.” *High-Performance Graphics Conference*.

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, Xuequan, Scott Schaefer, Jun Luo, Lizhuang Ma, and Ying He (2020). "Low Rank Matrix Approximation for 3D Geometry Filtering." *IEEE Transactions on Visualization and Computer Graphics (TVCG)*.

Lu, Xuequan, Shihao Wu, Honghua Chen, Sai-Kit Yeung, Wenzhi Chen, and Matthias Zwicker (2017). "GPF: GMM-inspired Feature-preserving Point Set Filtering." *IEEE Transactions on Visualization and Computer Graphics (TVCG)*.

Luo, Shitong and Wei Hu (2020). "Differentiable Manifold Reconstruction for Point Cloud Denoising." *ACM International Conference Multimedia*.

Luo, Shitong and Wei Hu (2021). "Score-Based Point Cloud Denoising." *IEEE International Conference on Computer Vision (ICCV)*.

Ma, Baorui, Zhizhong Han, Yu-Shen Liu, and Matthias Zwicker (2021). "Neural-Pull: Learning Signed Distance Functions from Point Clouds by Learning to Pull Space onto Surfaces." *International Conference on Machine Learning (ICML)*.

Macario Barros, Andréa, Maugan Michel, Yoann Moline, Gwenolé Corre, and Frédérick Carrel (2022). "A Comprehensive Survey of Visual SLAM Algorithms." *Robotics*.

Madaan, Pulkit, Abhishek Maiti, Saket Anand, and Sushil Mittal (2019). "Deep Mean Shift Clustering." *Bachelor's thesis. Indraprastha Institute of Information Technology*.

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

Maier, Robert, Raphael Schaller, and Daniel Cremers (2017b). "Efficient Online Surface Correction for Real-time Large-Scale 3D Reconstruction." *British Machine Vision Conference (BMVC)*.

Maimone, Andrew, Jonathan Bidwell, Kun Peng, and Henry Fuchs (2012). "Enhanced personal autostereoscopic telepresence system using commodity depth cameras." *Computers & Graphics*.

Maimone, Andrew and Henry Fuchs (2011). "Encumbrance-free Telepresence System with Real-time 3D Capture and Display Using Commodity Depth Cameras." *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*.

Maimone, Andrew and Henry Fuchs (2012). "Real-time volumetric 3D capture of room-sized scenes for telepresence." *3DTV-Conference*.

Mangiante, Simone, Guenter Klas, Amit Navon, Zhuang GuanHua, Ju Ran, and Marco Dias Silva (2017). “VR is on the Edge: How to Deliver 360° Videos in Mobile Networks.” *Workshop on Virtual Reality and Augmented Reality Network*.

Mao, Aihua, Zihui Du, Yu-Hui Wen, Jun Xuan, and Yong-Jin Liu (2022). “PD-Flow: A Point Cloud Denoising Framework with Normalizing Flows.” *arXiv preprint arXiv:2203.05940*.

Martin-Brualla, Ricardo, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth (2021). “NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections.” *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Mattei, Enrico and Alexey Castrodad (2017). “Point Cloud Denoising via Moving RPCA.” *Computer Graphics Forum (CGF)*.

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

Meka, Abhimitra, Maxim Maximov, Michael Zollhöfer, 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)*.

Mescheder, Lars, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger (2019). “Occupancy Networks: Learning 3D Reconstruction in Function Space.” *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Mildenhall, Ben, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng (2020). “NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis.” *European Conference on Computer Vision (ECCV)*.

Molyneaux, David, Shahram Izadi, David Kim, Otmar Hilliges, Steve Hodges, Xiang Cao, Alex Butler, and Hans Gellersen (2012). “Interactive Environment-Aware Handheld Projectors for Pervasive Computing Spaces.” *International Conference on Pervasive Computing*.

Mossel, Annette and Manuel Kroeter (2016). “Streaming and Exploration of Dynamically Changing Dense 3D Reconstructions in Immersive Virtual Reality.” *IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*.

Müller, Thomas, Alex Evans, Christoph Schied, and Alexander Keller (2022). “Instant Neural Graphics Primitives with a Multiresolution Hash Encoding.” *ACM Transactions on Graphics (TOG)*.

Mulligan, Jane and Kostas Daniilidis (2000). “View-independent scene acquisition for telepresence.” *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*.

Narváez, Esmeide A. Leal and Nallig Eduardo Leal Narváez (2006). “Point cloud denoising using robust principal component analysis.” *International Conference on Computer Graphics Theory and Applications*.

Natalini, Pierpaolo and Biagio Palumbo (2000). “Inequalities for the incomplete gamma function.” *Mathematical Inequalities and Applications*.

Newcombe, Richard A., Dieter Fox, and Steven M. Seitz (2015). “DynamicFusion: Reconstruction and Tracking of Non-rigid Scenes in Real-Time.” *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)*.

Ng, Edward W. and Murray Geller (1969). “A Table of Integrals of the Error Functions.” *Journal of Research of the National Bureau of Standards*.

Niemeyer, Michael, Jonathan T. Barron, Ben Mildenhall, Mehdi SM Sajjadi, Andreas Geiger, and Noha Radwan (2022). “RegNeRF: Regularizing Neural Radiance Fields for View Synthesis from Sparse Inputs.” *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

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

NVIDIA Corporation (2016). *CUDA Toolkit Documentation*. `https://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html`. Accessed: 2019-01-29.

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)*.

Ortiz, Joseph, Alexander Clegg, Jing Dong, Edgar Sucar, David Novotny, Michael Zollhoefer, and Mustafa Mukadam (2022). “iSDF: Real-Time Neural Signed Distance Fields for Robot Perception.” *Robotics: Science and Systems*.

Orts-Escolano, Sergio, Christoph Rhemann, Sean Fanello, Wayne Chang, Adarsh Kowdle, Yury Degtyarev, David Kim, Philip Davidson, Sameh Khamis, Mingsong Dou, Vladimir Tankovich, Charles Loop, Qin Cai, Philip Chou, Sarah Mennicken, Julien Valentin, Vivek Pradeep, Shenlong Wang, Sing Bing Kang, Pushmeet Kohli, Yuliya Lutchyn, Cem Keskin, and Shahram Izadi (2016). "Holoportation: Virtual 3D Teleportation in Real-time." *ACM Symposium on User Interface Software and Technology (UIST)*.

Osher, Stanley and James A. Sethian (1988). "Fronts Propagating with Curvature Dependent Speed: Algorithms Based on Hamilton-Jacobi Formulations." *Journal of Computational Physics*.

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

Pagh, Rasmus and Flemming Friche Rodler (2004). "Cuckoo Hashing." *Journal of Algorithms*.

Pagliari, Diana and Livio Pinto (2015). "Calibration of Kinect for Xbox One and Comparison between the Two Generations of Microsoft Sensors." *Sensors*.

Park, Jeong Joon, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove (2019). "DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Payne, Andrew, Andy Daniel, Anik Mehta, Barry Thompson, Cyrus S. Bamji, Dane Snow, Hideaki Oshima, Larry Prather, Mike Fenton, Lou Kordus, Pat O'Connor, Rich McCauley, Sheethal Nayak, Sunil Acharya, Swati Mehta, Tamer Elkhatib, Thomas Meyer, Tod O'Dwyer, Travis Perry, and Zhanping Xu (2014). "A 512×424 CMOS 3D Time-of-Flight Image Sensor with Multi-Frequency Photo-Demodulation up to 130MHz and 2GS/s ADC." *IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*.

Pejsa, Tomislav, Julian Kantor, Hrvoje Benko, Eyal Ofek, and Andrew Wilson (2016). "Room2Room: Enabling Life-Size Telepresence in a Projected Augmented Reality Environment." *ACM Conference on Computer-Supported Cooperative Work & Social Computing*.

Peng, Songyou, Chiyu Jiang, Yiyi Liao, Michael Niemeyer, Marc Pollefeys, and Andreas Geiger (2021). "Shape As Points: A Differentiable Poisson Solver." *Advanced Neural Information Processing Systems (NeurIPS)*.

Petit, Benjamin, Jean-Denis Lesage, Clément Menier, Jérémie Allard, Jean-Sébastien Franco, Bruno Raffin, Edmond Boyer, and François Faure (2010). "Multicamera Real-Time 3D Modeling for Telepresence and Remote Collaboration." *International Journal of Digital Multimedia Broadcasting*.

Pfister, Hanspeter, Matthias Zwicker, Jeroen Van Baar, and Markus Gross (2000). "Surfels: Surface Elements as Rendering Primitives." *Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*.

Pistilli, Francesca, Giulia Fracastoro, Diego Valsesia, and Enrico Magli (2020). "Learning Graph-Convolutional Representations for Point Cloud Denoising." *European Conference on Computer Vision (ECCV)*.

Pozzer, Cesar Tadeu, Cícero A. de Lara Pahins, and Ilona Heldal (2014). "A Hash Table Construction Algorithm for Spatial Hashing Based on Linear Memory." *Conference on Advances in Computer Entertainment Technology*.

Preiner, Reinhold, Oliver Mattausch, Murat Arikan, Renato Pajarola, and Michael Wimmer (2014). "Continuous Projection for Fast L_1 Reconstruction." *ACM Transactions on Graphics (TOG)*.

PresenterMedia (2009). *PowerPoint Templates, 3D Animations, and Clipart*. `https://presentermedia.com/`. Accessed: 2019-01-29.

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 R., Li Yi, Hao Su, and Leonidas J. Guibas (2017b). "PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space." *Advanced Neural Information Processing Systems (NeurIPS)*.

Rajput, Asif, Eugen Funk, Anko Börner, and Olaf Hellwich (2018). "A Regularized Volumetric Fusion Framework for Large-Scale 3D Reconstruction." *ISPRS Journal of Photogrammetry and Remote Sensing (P&RS)*.

Rakotosaona, Marie-Julie, Vittorio La Barbera, Paul Guerrero, Niloy J. Mitra, and Maks Ovsjanikov (2020). "PointCleanNet: Learning to Denoise and Remove Outliers from Dense Point Clouds." *Computer Graphics Forum (CGF)*.

Reichl, Florian, Jakob Weiss, and Rüdiger Westermann (2016). "Memory-Efficient Interactive Online Reconstruction From Depth Image Streams." *Computer Graphics Forum (CGF)*.

Remil, Oussama, Qian Xie, Xingyu Xie, Kai Xu, and Jun Wang (2017). "Surface Reconstruction with Data-driven Exemplar Priors." *Computer Aided Design*.

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)*.

Riegler, Gernot, Ali Osman Ulusoy, Horst Bischof, and Andreas Geiger (2017). "OctNetFusion: Learning Depth Fusion from Data." *International Conference on 3D Vision (3DV)*.

Rosman, Guy, Anastasia Dubrovina, and Ron Kimmel (2013). "Patch-Collaborative Spectral Point-Cloud Denoising." *Computer Graphics Forum (CGF)*.

Roth, Henry and Marsette Vona (2012). "Moving Volume KinectFusion." *British Machine Vision Conference (BMVC)*.

Roveri, Riccardo, A. Cengiz Öztireli, Ioana Pandele, and Markus Gross (2018). "PointProNets: Consolidation of Point Clouds with Convolutional Neural Networks." *Computer Graphics Forum (CGF)*.

Rückert, Darius and Marc Stamminger (2021). *International Symposium on Vision, Modeling, and Visualization (VMV)*.

Rusinkiewicz, Szymon, Olaf Hall-Holt, and Marc Levoy (2002). "Real-Time 3D Model Acquisition." *ACM Transactions on Graphics (TOG)*.

Sandström, Erik, Martin R. Oswald, Suryansh Kumar, Silvan Weder, Fisher Yu, Cristian Sminchisescu, and Luc Van Gool (2022). "Learning Online Multi-Sensor Depth Fusion." *arXiv preprint arXiv:2204.03353*.

Sasikumar, Prasanth, Lei Gao, Huidong Bai, and Mark Billinghurst (2019). "Wearable RemoteFusion: A Mixed Reality Remote Collaboration System with Local Eye Gaze and Remote Hand Gesture Sharing." *IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*.

Schilling, René L., Renming Song, and Zoran Vondracek (2012). *Bernstein Functions: Theory and Applications*.

Schoenberg, Isaac J. (1938). "Metric Spaces and Completely Monotone Functions." *Annals of Mathematics*.

Schöps, Thomas, Jakob Engel, and Daniel Cremers (2014). "Semi-Dense Visual Odometry for AR on a Smartphone." *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*.

Schöps, Thomas, Torsten Sattler, and Marc Pollefeys (2019a). "BAD SLAM: Bundle Adjusted Direct RGB-D SLAM." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Schöps, Thomas, Torsten Sattler, and Marc Pollefeys (2019b). "SurfelMeshing: Online Surfel-Based Mesh Reconstruction." *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*.

Schwartz, Christopher, Ralf Sarlette, Michael Weinmann, and Reinhard Klein (2013). "DOME II: A Parallelized BTF Acquisition System." *Material Appearance Modeling*.

Schwarz, Max, David Droeschel, Christian Lenz, Arul Selvam Periyasamy, En Yen Puang, Jan Razlaw, Diego Rodriguez, Sebastian Schüller, Michael Schreiber, and Sven Behnke (2019). "Team NimbRo at MBZIRC 2017: Autonomous valve stem turning using a wrench." *Journal of Field Robotics*.

Shi, Chunhao, Chunyang Wang, Xuelian Liu, Shaoyu Sun, Bo Xiao, Xuemei Li, and Guorui Li (2022). "Three-dimensional point cloud denoising via a gravitational feature function." *Applied Optics*.

Shi, Jian, Yue Dong, Xin Tong, and Yanyun Chen (2015). "Efficient Intrinsic Image Decomposition for RGBD Images." *ACM Symposium on Virtual Reality Software and Technology*.

Shun, Julian and Guy E. Blelloch (2014). "Phase-concurrent Hash Tables for Determinism." *ACM Symposium on Parallelism in Algorithms and Architectures*.

Stanford Computer Graphics Laboratory (1994). *The Stanford 3D Scanning Repository*. `https://graphics.stanford.edu/data/3Dscanrep/`. Accessed: 2022-09-21.

Stein, Elias M. and Guido Weiss (1971). *Introduction to Fourier Analysis on Euclidean Spaces*.

Steinbrücker, Frank, Christian Kerl, and Daniel Cremers (2013). "Large-Scale Multi-Resolution Surface Reconstruction from RGB-D Sequences." *IEEE International Conference on Computer Vision (ICCV)*.

Steinbrücker, Frank, Jürgen Sturm, and Daniel Cremers (2011). "Real-Time Visual Odometry from Dense RGB-D Images." *IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*.

Steinbrücker, Frank, Jürgen Sturm, and Daniel Cremers (2014). "Volumetric 3D Mapping in Real-Time on a CPU." *IEEE International Conference on Robotics and Automation (ICRA)*.

Stotko, Patrick (2015). "Improved 3D Reconstruction using Combined Weighting Strategies." *Central European Seminar on Computer Graphics for Students (CESCG)*.

Stotko, Patrick (2016a). "Interactive Appearance Reconstruction from RGB-D and IR data." *Master's thesis. University of Bonn*.

Stotko, Patrick (2016b). "State of the Art in Real-time Registration of RGB-D Images." *Central European Seminar on Computer Graphics for Students (CESCG)*.

Stotko, Patrick (2019a). "stdgpu: Efficient STL-like Data Structures on the GPU." *arXiv:1908.05936*. DOI: `10.48550/arXiv.1908.05936`.

Stotko, Patrick (2019b). *stdgpu: Efficient STL-like Data Structures on the GPU*. `https://github.com/stotko/stdgpu`. Accessed: 2022-09-21.

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)*. DOI: `10.1109/TVCG.2019.2899231`.

Stotko, Patrick, Stefan Krumpen, Reinhard Klein, and Michael Weinmann (2019b). "Towards Scalable Sharing of Immersive Live Telepresence Experiences Beyond Room-scale based on Efficient Real-time 3D Reconstruction and Streaming." *CVPR Workshop on Computer Vision for AR/VR*.

Stotko, Patrick, Stefan Krumpen, Max Schwarz, Christian Lenz, Sven Behnke, Reinhard Klein, and Michael Weinmann (2019c). "A VR System for Immersive Teleoperation and Live Exploration with a Mobile Robot." *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: `10.1109/IROS40897.2019.8968598`.

Stotko, Patrick, Stefan Krumpen, Michael Weinmann, and Reinhard Klein (2019d). "Efficient 3D Reconstruction and Streaming for Group-Scale Multi-Client Live Telepresence." *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. DOI: `10.1109/ISMAR.2019.00018`.

Stotko, Patrick, Michael Weinmann, and Reinhard Klein (2019e). "Albedo estimation for real-time 3D reconstruction using RGB-D and IR data." *ISPRS Journal of Photogrammetry and Remote Sensing (P&RS)*. DOI: `10.1016/j.isprsjprs.2019.01.018`.

Stotko, Patrick, Michael Weinmann, and Reinhard Klein (2022). "Incomplete Gamma Kernels: Generalizing Locally Optimal Projection Operators." *arXiv:2205.01087 (under review), submitted to IEEE Transactions of Pattern Analysis and Machine Intelligence (TPAMI)*. DOI: `10.48550/arXiv.2205.01087`.

Stückler, Jörg and Sven Behnke (2014). "Multi-Resolution Surfel Maps for Efficient Dense 3D Modeling and Tracking." *Journal of Visual Communication and Image Representation*.

Sturm, Jürgen, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers (2012). "A Benchmark for the Evaluation of RGB-D SLAM Systems." *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Sun, Yujing, Scott Schaefer, and Wenping Wang (2015). "Denoising point sets via L_0 minimization." *Computer Aided Geometric Design*.

Taketomi, Takafumi, Hideaki Uchiyama, and Sei Ikeda (2017). "Visual SLAM algorithms: a survey from 2010 to 2016." *IPSJ Transactions on Computer Vision and Applications*.

Tanikawa, Tomohiro, Yasuhiro Suzuki, Koichi Hirota, and Michitaka Hirose (2005). "Real World Video Avatar: Real-time and Real-size Transmission and Presentation of Human Figure." *International Conference on Augmented Tele-existence*.

Tanke, Julian, Oh-Hun Kwon, Patrick Stotko, Radu Alexandru Rosu, Michael Weinmann, Hassan Errami, Sven Behnke, Maren Bennewitz, Reinhard Klein, Andreas Weber, Angela Yao, and Juergen Gall (2019). "Bonn Activity Maps: Dataset Description." *arXiv:1912.06354*. DOI: `10.48550/arXiv.1912.06354`.

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)*.

Teschner, Matthias, Bruno Heidelberger, Matthias Müller, Danat Pomerantes, and Markus Gross (2003). "Optimized Spatial Hashing for Collision Detection of Deformable Objects." *International Symposium on Vision, Modeling, and Visualization (VMV)*.

Tewari, Ayush, Justus Thies, Ben Mildenhall, Pratul Srinivasan, Edgar Tretschk, Yifan Wang, Christoph Lassner, Vincent Sitzmann, Ricardo Martin-Brualla, Stephen Lombardi, Tomas Simon, Christian Theobalt, Matthias Nießner, Jonathan T. Barron, Gordon Wetzstein, Michael Zollhöfer, and Vladislav Golyanik (2022). "Advances in Neural Rendering." *Computer Graphics Forum (CGF)*.

Tomasi, Carlo and Roberto Manduchi (1998). "Bilateral Filtering for Gray and Color Images." *IEEE International Conference on Computer Vision (ICCV)*.

Towles, Herman, Wei-Chao Chen, Ruigang Yang, Sang-Uok Kum, Henry Fuchs Nikhil Kelshikar, Jane Mulligan, Kostas Daniilidis, Henry Fuchs, Carolina Chapel Hill, Nikhil Kelshikar Jane Mulligan, et al. (2002). "3D Tele-Collaboration Over Internet2." *International Workshop on Immersive Telepresence*.

Tran, Tuan Tu, Mathieu Giraud, and Jean-Stéphane Varré (2015). "Perfect Hashing Structures for Parallel Similarity Searches." *IEEE International Parallel and Distributed Processing Symposium Workshop*.

University of Bonn (2021). *Wie virtuelle Realität real wird*. `https://www.uni-bonn.de/de/forschung-lehre/transfercenter-enacom/news/wie-virtuelle-realitaet-real-wird`. Accessed: 2022-09-21.

Valgma, Lembit (2016). "3D reconstruction using Kinect v2 camera." *Bachelor's thesis*. *University of Tartu*.

Vasudevan, Ramanarayan, Gregorij Kurillo, Edgar Lobaton, Tony Bernardin, Oliver Kreylos, Ruzena Bajcsy, and Klara Nahrstedt (2011). "High-Quality Visualization for Geographically Distributed 3-D Teleimmersive Applications." *IEEE Transactions on Multimedia*.

Vincent, Pascal, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol (2008). “Extracting and Composing Robust Features with Denoising Autoencoders.” *International Conference on Machine Learning (ICML)*.

Wang, Zixiong, Pengfei Wang, Qiujie Dong, Junjie Gao, Shuangmin Chen, Shiqing Xin, and Changhe Tu (2021). “Neural-IMLS: Learning Implicit Moving Least-Squares for Surface Reconstruction from Unoriented Point Clouds.” *arXiv preprint arXiv:2109.04398*.

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

Weder, Silvan, Johannes L. Schönberger, Marc Pollefeys, and Martin R. Oswald (2020). “RoutedFusion: Learning Real-time Depth Map Fusion.” *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Weder, Silvan, Johannes L. Schönberger, Marc Pollefeys, and Martin R. Oswald (2021). “NeuralFusion: Online Depth Fusion in Latent Space.” *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Weiszfeld, Endre (1937). “Sur le point pour lequel la somme des distances de n points donnés est minimum.” *Tohoku Mathematical Journal*.

Wendland, Holger (1995). “Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree.” *Advances in Computational Mathematics*.

Whelan, Thomas, Hordur Johannsson, Michael Kaess, John J. Leonard, and John McDonald (2013). “Robust Real-Time Visual Odometry for Dense RGB-D Mapping.” *IEEE International Conference on Robotics and Automation (ICRA)*.

Whelan, Thomas, Michael Kaess, Maurice Fallon, Hordur Johannsson, John J. Leonard, and John McDonald (2012). “Kintinuous: Spatially Extended KinectFusion.” *RSS Workshop on RGB-D: Advanced Reasoning with Depth Cameras*.

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

Whelan, Thomas, Stefan Leutenegger, Renato F. Salas-Moreno, Ben Glocker, and Andrew J. Davison (2015b). “ElasticFusion: Dense SLAM Without A Pose Graph.” *Robotics: Science and Systems*.

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)*.

Witmer, Bob G. and Michael J. Singer (1998). "Measuring Presence in Virtual Environments: A Presence Questionnaire." *Presence: Teleoperators and Virtual Environments*.

Wu, Chenglei, Michael Zollhöfer, Matthias Nießner, Marc Stamminger, Shahram Izadi, and Christian Theobalt (2014). "Real-time Shading-based Refinement for Consumer Depth Cameras." *ACM Transactions on Graphics (TOG)*.

Wu, Hongzhi, Zhaotian Wang, and Kun Zhou (2016). "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)*.

Xu, Xueli, Guohua Geng, Xin Cao, Kang Li, and Mingquan Zhou (2022a). "TDNet: transformer-based network for point cloud denoising." *Applied Optics*.

Xu, Yabin, Liangliang Nan, Laishui Zhou, Jun Wang, and Charlie C. L. Wang (2022b). "HRBF-Fusion: Accurate 3D Reconstruction from RGB-D Data Using On-the-fly Implicits." *ACM Transactions on Graphics (TOG)*.

Xu, Zhenhua, Jiancheng Lai, Jun Zhou, Huasong Chen, Hongkun Huang, and Zhenhua Li (2022c). "Image Deblurring Using a Robust Loss Function." *Circuits, Systems, and Signal Processing*.

Yadav, Sunil Kumar, Martin Skrodzki, Eric Zimmermann, and Konrad Polthier (2021). "Surface Denoising based on Normal Filtering in a Robust Statistics Framework." *Forum "Math-for-Industry" 2018*.

Yamasaki, Ryoya and Toshiyuki Tanaka (2019). "Properties of Mean Shift." *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*.

Yan, Zike, Yuxin Tian, Xuesong Shi, Ping Guo, Peng Wang, and Hongbin Zha (2021). "Continual Neural Mapping: Learning An Implicit Scene Representation from Sequential Observations." *IEEE International Conference on Computer Vision (ICCV)*.

Yifan, Wang, Felice Serena, Shihao Wu, A. Cengiz Öztireli, and Olga Sorkine-Hornung (2019a). "Differentiable Surface Splatting for Point-based Geometry Processing." *ACM Transactions on Graphics (TOG)*.

Yifan, Wang, Shihao Wu, Hui Huang, Daniel Cohen-Or, and Olga Sorkine-Hornung (2019b). "Patch-based Progressive 3D Point Set Upsampling." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Young, Jacob, Tobias Langlotz, Steven Mills, and Holger Regenbrecht (2020). "Mobileportation: Nomadic Telepresence for Mobile Devices." *ACM Interactive, Mobile, Wearable and Ubiquitous Technologies.*

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).*

Yuan, Yu-Jie, Yang-Tian Sun, Yu-Kun Lai, Yuewen Ma, Rongfei Jia, and Lin Gao (2022). "NeRF-Editing: Geometry Editing of Neural Radiance Fields." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Zaman, Faisal, Ya Ping Wong, and Boon Yian Ng (2017). "Density-Based Denoising of Point Cloud." *International Conference on Robotics, Vision, Signal Processing and Power Applications.*

Zeng, Jin, Gene Cheung, Michael Ng, Jiahao Pang, and Cheng Yang (2019). "3D Point Cloud Denoising Using Graph Laplacian Regularization of a Low Dimensional Manifold Model." *IEEE Transactions on Image Processing.*

Zeng, Ming, Fukai Zhao, Jiaxiang Zheng, and Xinguo Liu (2012). "A memory-efficient kinectfusion using octree." *International Conference on Computational Visual Media.*

Zennaro, Simone, Matteo Munaro, Simone Milani, Pietro Zanuttigh, Andrea Bernardi, Stefano Ghidoni, and Emanuele Menegatti (2015). "Performance evaluation of the 1st and 2nd generation Kinect for multimedia applications." *IEEE International Conference on Multimedia and Expo (ICME).*

Zhang, Di, Feng Xu, Chi-Man Pun, Yang Yang, Rushi Lan, Liejun Wang, Yujie Li, and Hao Gao (2021a). "Virtual Reality Aided High-Quality 3D Reconstruction by Remote Drones." *ACM Transactions on Internet Technology.*

Zhang, Dongbo, Xuequan Lu, Hong Qin, and Ying He (2020). "Pointfilter: Point Cloud Filtering via Encoder-Decoder Modeling." *IEEE Transactions on Visualization and Computer Graphics (TVCG).*

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

Zhang, Juyong, Bailin Deng, Yang Hong, Yue Peng, Wenjie Qin, and Ligang Liu (2018). "Static/Dynamic Filtering for Mesh Geometry." *IEEE Transactions on Visualization and Computer Graphics (TVCG)*.

Zhang, Xiaoshuai, Sai Bi, Kalyan Sunkavalli, Hao Su, and Zexiang Xu (2022b). "NeRFusion: Fusing Radiance Fields for Large-Scale Scene Reconstruction." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zhang, Xiuming, Pratul P. Srinivasan, Boyang Deng, Paul Debevec, William T. Freeman, and Jonathan T. Barron (2021b). "NeRFactor: Neural Factorization of Shape and Reflectance Under an Unknown Illumination." *ACM Transactions on Graphics (TOG)*.

Zhang, Zhengyou (2000). "A Flexible New Technique for Camera Calibration." *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*.

Zhao, Yaping, Haitian Zheng, Zhongrui Wang, Jiebo Luo, and Edmund Y. Lam (2022). "Point Cloud Denoising via Momentum Ascent in Gradient Fields." *arXiv preprint arXiv:2202.10094*.

Zheng, Yinglong, Guiqing Li, Shihao Wu, Yuxin Liu, and Yuefang Gao (2017). "Guided point cloud denoising via sharp feature skeletons." *The Visual Computer*.

Zhou, Hang, Kejiang Chen, Weiming Zhang, Han Fang, Wenbo Zhou, and Nenghai Yu (2019). "DUP-Net: Denoiser and Upsampler Network for 3D Adversarial Point Clouds Defense." *IEEE International Conference on Computer Vision (ICCV)*.

Zhou, Qian-Yi and Vladlen Koltun (2013). "Dense Scene Reconstruction with Points of Interest." *ACM Transactions on Graphics (TOG)*.

Zhou, Qian-Yi and Vladlen Koltun (2014). "Color Map Optimization for 3D Reconstruction with Consumer Depth Cameras." *ACM Transactions on Graphics (TOG)*.

Zhou, Qian-Yi, Jaesik Park, and Vladlen Koltun (2018). "Open3D: A Modern Library for 3D Data Processing." *arXiv preprint arXiv:1801.09847*.

Zhu, Dingkun, Honghua Chen, Weiming Wang, Haoran Xie, Gary Cheng, Mingqiang Wei, Jun Wang, and Fu Lee Wang (2022). "Non-local Low-rank Point Cloud Denoising for 3D Measurement Surfaces." *IEEE Transactions on Instrumentation and Measurement*.

Zillner, Jakob, Erick Mendez, and Daniel Wagner (2018). "Augmented Reality Remote Collaboration with Dense Reconstruction." *IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*.

Zingsheim, Domenic, Patrick Stotko, Stefan Krumpen, Michael Weinmann, and Reinhard Klein (2021). "Collaborative VR-based 3D Labeling of Live-captured Scenes by Remote Users." *IEEE Computer Graphics and Applications (CG&A)*. DOI: `10.1109/MCG.2021.3082267`.

Zollhöfer, Michael, Patrick Stotko, Andreas Görlitz, Christian Theobalt, Matthias Nießner, Reinhard Klein, and Andreas Kolb (2018). "State of the Art on 3D Reconstruction with RGB-D Cameras." *Computer Graphics Forum (CGF)*. DOI: `10.1111/cgf.13386`.

Zumbach, Gilles and Ulrich Müller (2001). "Operators on inhomogeneous time series." *International Journal of Theoretical and Applied Finance*.

Zuo, Xinxin, Sen Wang, Jiangbin Zheng, and Ruigang Yang (2017). "Detailed Surface Geometry and Albedo Recovery from RGB-D Video under Natural Illumination." *IEEE International Conference on Computer Vision (ICCV)*.

List of Figures

List of Tables

Part IV

Appendix

Publication:
“SLAMCast: Large-Scale, Real-Time 3D Reconstruction and Streaming for Immersive Multi-Client Live Telepresence”

Patrick Stotko, Stefan Krumpen, Matthias B. Hullin,
Michael Weinmann, and Reinhard Klein

IEEE Transactions on Visualization and Computer Graphics (TVCG)

2019

DOI: 10.1109/TVCG.2019.2899231

SLAMCast: Large-Scale, Real-Time 3D Reconstruction and Streaming for Immersive Multi-Client Live Telepresence

Patrick Stotko, Stefan Krumpen, Matthias B. Hullin, Michael Weinmann, and Reinhard Klein

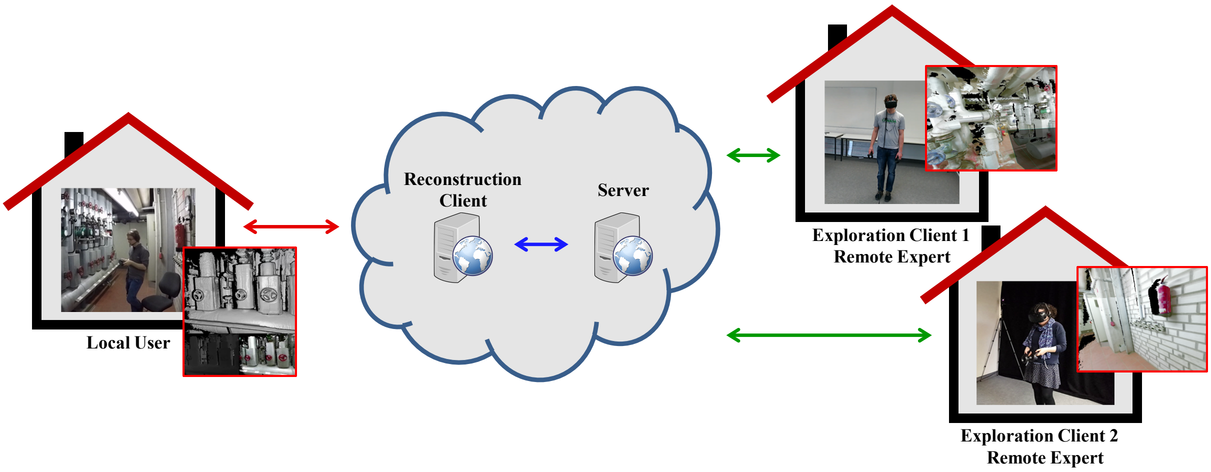


Fig. 1. Illustration of our novel multi-client live telepresence framework for remote collaboration: RGB-D data captured with consumer-grade cameras represent the input to our real-time large-scale reconstruction technique that is based on a novel thread-safe GPU hash map data structure. Efficient data streaming is achieved by transmitting a novel compact representation of the reconstructed model in terms of Marching Cubes indices. Multi-client live telepresence is achieved by the server's independent handling of client requests.

Abstract— Real-time 3D scene reconstruction from RGB-D sensor data, as well as the exploration of such data in VR/AR settings, has seen tremendous progress in recent years. The combination of both these components into telepresence systems, however, comes with significant technical challenges. All approaches proposed so far are extremely demanding on input and output devices, compute resources and transmission bandwidth, and they do not reach the level of immediacy required for applications such as remote collaboration. Here, we introduce what we believe is the first practical client-server system for real-time capture and many-user exploration of static 3D scenes. Our system is based on the observation that interactive frame rates are sufficient for capturing and reconstruction, and real-time performance is only required on the client site to achieve lag-free view updates when rendering the 3D model. Starting from this insight, we extend previous voxel block hashing frameworks by introducing a novel thread-safe GPU hash map data structure that is robust under massively concurrent retrieval, insertion and removal of entries on a thread level. We further propose a novel transmission scheme for volume data that is specifically targeted to Marching Cubes geometry reconstruction and enables a 90% reduction in bandwidth between server and exploration clients. The resulting system poses very moderate requirements on network bandwidth, latency and client-side computation, which enables it to rely entirely on consumer-grade hardware, including mobile devices. We demonstrate that our technique achieves state-of-the-art representation accuracy while providing, for any number of clients, an immersive and fluid lag-free viewing experience even during network outages.

Index Terms—Remote collaboration, live telepresence, real-time reconstruction, voxel hashing, RGB-D, real-time streaming.

1 Introduction

One of the main motivations behind virtual reality research has always been to allow users to immersively and subjectively explore remote places or environments. An experience of telepresence could benefit applications as diverse as remote collaboration, entertainment, advertisement, teaching, hazard site exploration, or rehabilitation. Thanks to advances in display technology and the emergence of high-resolution head-mounted devices, we have seen a recent surge in virtual reality solutions. However, it has long been known that traditional display parameters like resolution, frame rate and contrast are not the only factors contributing to an immersive viewing experience. The presentation of the data, its consistency, low-latency control to avoid motion sickness, the degree of awareness and the suitability of controller devices are just as important [11, 16, 52]. For applications such as remote exploration, remote collaboration or teleconferencing, these conditions are not easily met, as the scene is not pre-built but needs to be reconstructed on-the-fly from 3D input data captured by a person or robotic device. At the same time, the data flow in a well-designed system should give multiple remote users the freedom to individually explore, for instance using head-mounted displays (HMD), the current state of reconstruction in the most responsive way possible.

A particular challenge, therefore, is to find a suitable coupling between the acquisition and viewing stages that respects the practical limitations imposed by available network bandwidth and client-side compute hardware while still guaranteeing an immersive exploration experience. For this purpose, teleconferencing systems for transmit-

- *Patrick Stotko, Stefan Krumpen, Matthias B. Hullin, Michael Weinmann, and Reinhard Klein are with University of Bonn. E-mail: {stotko, krumpen, hullin, mw, rk}@cs.uni-bonn.de.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

ting dynamic 3D models of their *users* typically rely on massive well-calibrated acquisition setups with several statically mounted cameras around the region of interest [9, 40, 49]. Instead, we direct our attention to the remote exploration of *places* using portable, consumer-grade acquisition devices, for instance in scenarios of remote inspection or consulting. On the acquisition site, a user digitizes their physical environment using consumer-grade 3D capture hardware. Remote clients can perform immersive and interactive live inspection of that environment using off-the-shelf VR devices even while it is acquired and progressively refined. In this scenario, additional challenges arise as the incoming amount of captured data may be high and may also significantly vary over time depending on the size of the scene that is currently imaged. The latter particularly happens for strongly varying object distances within the captured data, whereas the amount of data over time remains in the same order of magnitude if the objects are within the same distance to the capturing camera (as met for teleconferencing scenarios). A first attempt towards interactive virtual live inspection of real scenes [35] built upon real-time voxel block hashing based 3D reconstruction [22] using implicit truncated signed distance fields (TSDFs) that has become a well-established method for high-quality reconstructions [8, 19, 22, 37–39, 50]. Voxel blocks that are completely processed, i.e. those that are no longer visible, are immediately sent to the remote client and locally converted into a mesh representation using Marching Cubes [30] to perform the actual rendering. Besides the fact that the system is restricted to one remote user, other limitations are the rather high bandwidth requirement of up to 175MBit/s and the missing handling of network failures where the remote client has to reconnect. In particular for multi-client scenarios, handling both the bandwidth problem and the reconnection problem is of utmost importance to allow a satisfactory interaction between the involved users.

To overcome these problems, we propose a novel efficient low-cost multi-client remote collaboration system for the exploration of quasi-static scenes that is designed as a scalable client-server system which handles an arbitrary number of exploration clients under real-world network conditions (including the recovery from full outages) and using consumer-grade hardware. The system consists of a voxel block hashing based reconstruction client, a server managing the reconstructed model and the streaming states of the connected clients as well as the exploration clients themselves (see Fig. 1). The realization of the system relies on the following two key innovations:

- A novel scene representation and transmission protocol based on Marching Cubes (MC) indices enables the system to operate in low-bandwidth remote connection scenarios. Rather than reconstructing geometry on the server site or even performing server-side rendering, our system encodes the scene as a compressed sequence of voxel block indices and values, leaving the final geometry reconstruction to the exploration client. This results in significantly reduced bandwidth requirements compared to previous voxel based approaches [35].
- For the scalable, reliable and efficient management of the streaming states of the individual exploration clients, we propose a novel thread-leveled GPU hash set and map datastructure that guarantees successful concurrent retrieval, insertion and removal of millions of entries on the fly while preserving key uniqueness without any prior knowledge about the data.

From a system point of view, the extension of the system towards multiple reconstruction clients [15] is also envisioned but beyond the scope of this paper. In order to overcome the inherently limited resolution of voxel-based scene representations, we also include a lightweight projective texture mapping approach that enables the visualization of texture details at the full resolution of the depth camera on demand. Users collaboratively exploring the continuously captured scene experience a strong telepresence effect and are directly able to start conversation about the distant environment. We motivate the need of a client server system, provide a discussion of the respective challenges and design choices, and evaluate the proposed system regarding latency, visual quality and accuracy. Furthermore, we demonstrate its practicality in a multi-client remote servicing and inspection role-play scenario with non-expert users (see supplemental video).

2 Related Work

In this section, we provide an overview of previous efforts related to our novel large-scale, real-time 3D reconstruction and streaming framework for immersive multi-client telepresence categorized according to the developments regarding telepresence, 3D reconstruction and hashing.

2.1 Telepresence

Real-time 3D reconstruction is a central prerequisite for many immersive telepresence applications. Early multi-camera telepresence systems did not allow the acquisition and transmission of high-quality 3D models in real-time to remote users due to limitations regarding the hardware at the time [12, 24, 27, 36, 46, 47] or the applied techniques such as the lacking reconstruction accuracy of shape-from-silhouette approaches for concave surface regions [29, 41]. Then the spreading access to affordable commodity depth sensors such as the Microsoft Kinect led to the development of several 3D reconstruction approaches at room scale [13, 19, 20, 31, 32, 34]. However, the high sensor noise as well as temporal inconsistency in the reconstruction limited the quality of the reconstructions. Furthermore, Photoportals [26] have been proposed to provide immersive access to pre-captured 3D virtual environments while also supporting remote collaborative exploration. However, including live-captured contents comes at the cost of a significant lag as well as a reduced resolution. In contrast, the Holoportation system [40] is built on top of the accurate real-time 3D reconstruction pipeline Fusion4D [8] and involves real-time data transmission as well as AR and VR technology to achieve an end-to-end immersive teleconferencing experience. However, massive hardware requirements, i.e. several high-end GPUs running on multiple desktop computers, were needed to achieve real-time performance, where most of the expensive hardware components need to be located at the local user's side. In the context of static scene telepresence, Mossel and Kröter [35] developed an interactive single-exploration-client VR application based on current voxel block hashing techniques [22]. Although the system is restricted to only one exploration client, the bandwidth requirements of this approach have been reported to be up to 175MBit/s in a standard scenario. A further issue resulting from the direct transmission of the captured data to the rendering client occurs in case of network interruptions where the exploration client has to reconnect to the reconstruction client. Since the system does not keep track of the transmitted data, parts of the scene that are reconstructed during network outage will be lost. While previous approaches are only designed for single client telepresence or do not support interactive collaboration, our approach overcomes these limitations and enables a variety of new applications.

2.2 3D Reconstruction

The key to success of the recently emerging high-quality real-time reconstruction frameworks is the underlying data representation that is used to fuse the incoming sensor measurements. Especially the modeling of surfaces in terms of implicit truncated signed distance fields (TSDFs) has become well-established for high-quality reconstructions. Earlier of these volumetric reconstruction frameworks such as KinectFusion [19, 37] rely on the use of a uniform grid so that the memory requirement linearly scales with the overall grid size and not with the significantly smaller subset of surface areas. As this is impractical for handling large-scale scenes, follow-up work focused on the development of efficient data structures for real-time volumetric data fusion by exploiting sparsity in the TSDF. This has been achieved based on using moving volume techniques [43, 51], representing scenes in terms of blocks of volumes that follow dominant planes [17] or height maps that are parameterized over planes [44], or using dense volumes only in the vicinity of the actual surface areas to store the TSDF [4, 22, 39]. The allocated blocks that need to be indexed may be addressed based on tree structures or hash maps. Tree structures model the spatial hierarchy at the cost of a complex parallelization and a time-consuming tree traversal which can be avoided with the use of hash functions that, however, discard the hierarchy. Nießner et al. [39] proposed real-time 3D reconstruction based on a spatial voxel block hashing framework that has been later optimized [22]. Drift that may lead to the accumulation of errors in the reconstructed model [39] can be counteracted by

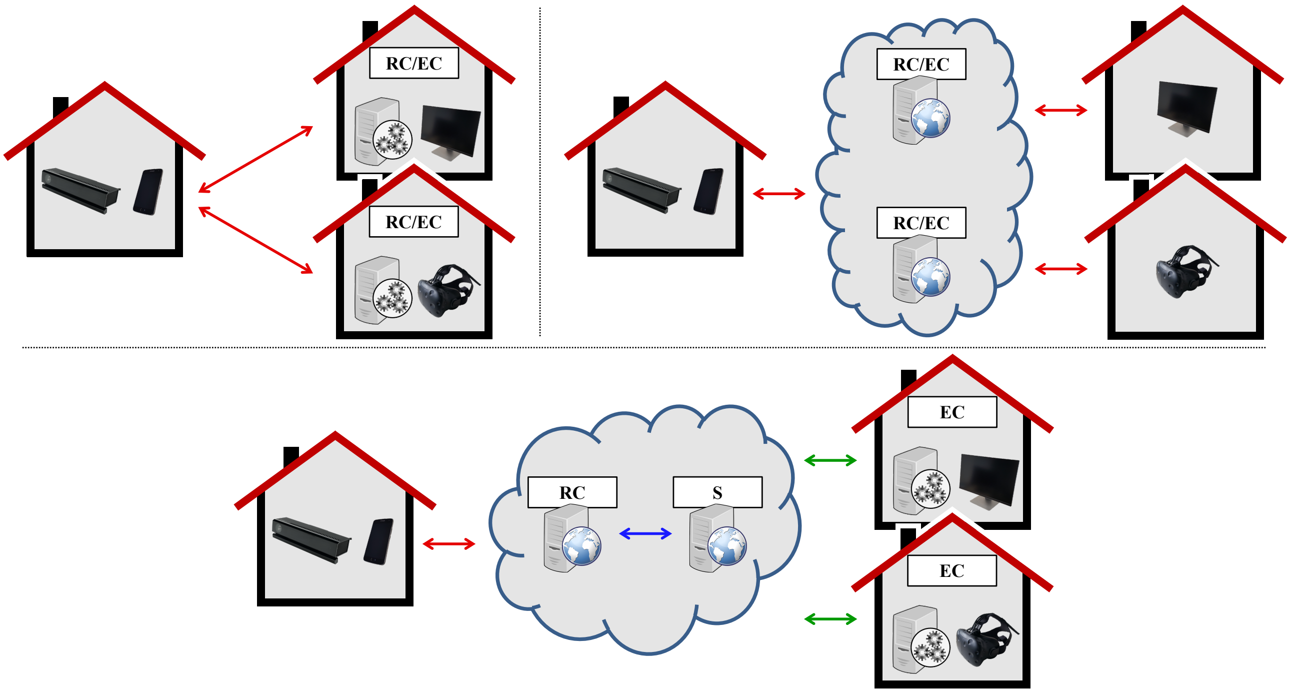


Fig. 2. Possible design choices regarding the architecture of an end-to-end VR collaboration system. Although both the reconstruction client (RC) and the exploration client (EC) can be realized at the remote expert's side or inside the cloud (top row) to rely on standard video streaming techniques (red arrows), such systems either impose an extremely high computational burden to the client's machine (top left) or fail to provide an immersive VR experience due to Internet latencies. Our system (bottom) overcomes these limitations by streaming the reconstructed 3D model to the individual exploration clients using a novel compact and bandwidth-optimized representation (green arrows). Note, that the use of multiple reconstruction clients can be naturally realized in this setting by transmitting the original representation (blue arrows) between reconstruction client (RC) and server (S).

implementing loop closure [7,21]. Due to its efficiency, we built our remote collaboration system on top of the voxel block hashing approach and adapt the latter to the requirements discussed before. Very recently, Golodetz et al. [15] presented a system for multi-client collaborative acquisition and reconstruction of static scenes with smartphones. For each connected camera, a submap representing the client-specific scene part is reconstructed and managed by a server. After capturing has finished, all submaps are merged into a final globally consistent 3D model to avoid artifacts arising from non-perfectly matching submap borders [21]. In contrast we focus on the development of a practical collaboration system for on-the-fly scene inspection and interaction by an arbitrary number of exploration clients. In this scenario, issues such as the submap jittering caused by progressive relocalization during the capturing process have to be handled carefully in order to preserve an acceptable VR experience. As the respective adequate adjustment of the submaps has to be evaluated in the scope of comprehensive user studies, we consider this challenge to be beyond the scope of this paper.

2.3 Hashing

Lossless packing of sparse data into a dense map can be achieved via hashing. However, developing such data structures on the GPU offering the reliability of their CPU-side counterparts is highly challenging. Current voxel block hashing techniques [7,22,39] including hierarchical voxel block hashing [23] rely on the high camera frame rate to clean up block allocation failures in subsequent frames and, thus, guarantee consistent but not necessarily successful insertion and removal. Only the guarantee regarding key uniqueness is strictly enforced to avoid that duplicate blocks are allocated and integrated during fusion. Although data integration for some voxel blocks (and re-integration [7]) might, hence, be staggered to a few subsequent frames, model consistency is still ensured by the high frame rate fusion. To achieve a more reliable GPU hashing, perfect hashing approaches [3,28,48] have been proposed that aim at collision-free hashing, but are hardly applicable for online reconstruction. In the context of collision handling, minimizing the maximum age of the hash map, i.e. the maximum number of required lookups during retrieval, by reordering key-value pairs similar to Cuckoo Hashing improves the robustness of the hash map construction [14]. Similar to Alcantara et al. [1], who analyzed different collision resolving strategies, the entry size is restricted to 64-bit due to the limited support size of atomic exchange operations. However, these approaches do not support entry removal and insertion is allowed to fail in case the defined upper bound on the maximum age is not achieved. Stadium Hashing [25] supports concurrent insertion and retrieval, but lacks removal, by avoiding entry reordering that would otherwise lead to synchronization issues. Recently, Ashkinani et al. [2] presented a fully dynamic hash map supporting concurrent insertion, retrieval, and also removal based chaining to resolve collisions. However, their data structure cannot enforce key uniqueness, which is an essential property required by voxel block hashing frameworks to preserve model consistency. In contrast, our hash map data structure overcomes all of the aforementioned limitations and is specifically suited for continuously updated reconstruction and telepresence scenarios.

3 Design Choices

In a practical remote communication and collaboration system, users should be able to directly start a conversation about the – possibly very large – environment/scene and experience an immersive live experience without the need for time-consuming prerecording similar to a telephone call. Such systems rely on efficient data representation and processing (see Table 1), immediate transmission as well as fast and compact data structures to allow reconstructing and providing a virtual 3D model in real time to remote users. In order to meet the requirements regarding usability, latency, and stability, several crucial design choices have to be taken into account. In particular, we thus focus on the discussion of a system design that benefits a variety of applications, while allowing the distribution of the computational burden according to the hardware availability respectively, i.e. to the cloud or to the remote expert's equipment, and scaling to many remote clients.

Table 1. Advantages and disadvantages of different scene representations for remote collaboration systems.

Data Representation	Flexibility	Individual Exploration	Re-Connection	Data Management	Compactness
RGB-D Data	-	-	-	easy	good
Voxel Block Model	✓	✓	✓	easy	bad
Mesh	✓	✓	✓	hard	good
MC index based Model	✓	✓	✓	easy	very good

Naïve Input Video Streaming An obvious strategy for the interactive exploration of a live-captured scene by the user is the transmission of the RGB-D input sequence and the reconstruction of the scene model at the exploration client's site (see Fig. 2 top left). Whereas the current state of the art in image and video compression techniques as well as real-time reconstruction would certainly be sufficient for the development of such systems, this approach has several limitations. First, such a setup imposes an extremely high computational burden to the remote expert's machine, where both the reconstruction and the rendering have to be performed, such that a smooth VR experience at 90Hz may not be guaranteed. Furthermore, in case of network outages, parts of the scene that are acquired while the exploration client is disconnected cannot be recovered automatically and the local user performing the capturing of the scene is forced to move back and acquire the missing parts again. In the worst case where the exploration client completely looses the currently reconstructed model, e.g. when the user accidentally closes the client, the whole capturing session must be restarted. In contrast, this problem can be avoided by instead streaming parts of the fused 3D model where the streaming order is not limited to the acquisition order and can, thus, be controlled for each exploration client independently according to their particular interests.

Full Cloud Video Streaming Alternatively, the full reconstruction including triangulation could be performed on a central cloud server and only RGB-D video streams are transmitted from/to the users (see Fig. 2 top right). While re-connections do not require further handling and data loss is no longer an issue, however, Internet latency becomes an apparent problem and prohibits an immersive VR experience. Lags in transmitting the video data directly affect the user experience. Standard approaches trying to compensate this issue rely on the view-adapted transmission of 360 degree video data (e.g. [6, 10, 18]). This allows inspecting the scene based on head rotations, however, translations through the scene are not supported. Furthermore, this not only requires that the users do not perform any fast movements, but also results in drastically increased bandwidth requirements due to the transmission of 360 degree video data which can easily result in the range of around 100MBit/s for SD video at 30Hz or more than 1GBit/s for 4K resolution at 120Hz respectively [33] which is higher than streaming the 3D model. The additional use of techniques for view specification based on e.g. fixation prediction [10] result in additional delays of around 40ms which represents a noticeable perceivable lag in remote collaboration scenarios and reduces the interactive experience. In addition, when the reconstruction is finished or paused and the 3D model does not change for a certain time, the video stream of the renderings still requires a constantly high amount of bandwidth whereas the bandwidth required for streaming the 3D model would immediately drop to zero.

Mesh Data Streaming When deciding for the aforementioned server architecture, there remains still the question which data should be transferred from the server to the exploration clients. Similar to full cloud-based video streaming, mesh updates could be streamed to the exploration clients and directly rendered at their machines using standard graphics APIs. Whereas the mesh representation is more compact in comparison to the voxel block model that is used for reconstruction, the number of triangles in each updated block largely differs depending on the amount of surface inside resulting in significantly more complicated and less efficient data management, updating and transmission. Furthermore, the vertex positions, which are given in the global coordinate system, are much harder to compress due to their irregular and arbitrary bit pattern. Instead, we propose a novel bandwidth-optimized representation based on Marching Cubes indices (see Sect. 4.2) that is even more compact after compression due to its more regular nature.

Centralized Data Processing We focus on the development of a system that is particularly designed for collaboration tasks where users can explore and interact with the captured scene while at the same time being able to observe the other client's interactions. For this purpose, a central server is placed between the individual clients to simplify the communication between clients and move shared computational work away from the clients. Using a server avoids complicated and error-prone dense mesh networks between all the exploration clients. Furthermore, it naturally facilitates the integration of multiple reconstruction clients and it allows lower hardware requirements at the exploration clients. This, in turn, makes the system suitable for a much broader variety of users. Powerful hardware, required for the scalability to a large number of clients, can be provided as practical cloud services or similar services (see Fig. 2 bottom).

Hash Data Structure Efficient data structures are crucial for efficiently and reliably managing the set of updated blocks for each connected exploration client as well as the scene model and therefore have to be adequately taken into account during the design phase. For data management, fast and efficient retrieval of subsets as well as guaranteed modification through duplicate-free insertion and deletion, which both implicitly perform retrieval to ensure uniqueness, are strictly required to avoid data loss during transmission or redundant streaming of data. In particular, the streaming states of each connected client, i.e. the set of updated data that needs to be transmitted, must be maintained in real-time to avoid delays during live exploration. Since the support for re-connections is a major feature of our telepresence system, these states will contain the list of blocks updated in the time while the connection was down or all blocks in case the client was closed accidentally by the user. Selecting a subset (which involves retrieval and deletion) as well as filling the state (which should be duplicate-free to avoid redundant transmissions) should, hence, be performed as fast as possible in parallel on the GPU for which hash data structures are highly suitable and have been well-established (e.g. [22, 39]). While recently developed hashing approaches work well with high-frame-rate online 3D reconstruction techniques, their lack of strong guarantees regarding hash operations make them hardly applicable to use cases with high reliability requirements such as telepresence systems. Dispensing with the uniqueness guarantee would lead to redundantly transmitted data and, hence, wasted bandwidth whereas artifacts such as holes will occur when insertion, removal, and retrieval cannot be guaranteed and these blocks get lost during streaming from the reconstruction client until the exploration client. With a novel hash map data structure that supports concurrent insertion, removal, and retrieval including key uniqueness preservation while running on a thread level, we directly address these requirements. A detailed evaluation regarding run time and further relevant design choices are provided in the supplemental material.

4 Proposed Remote Collaboration System

The overall server-client architecture of our novel framework for efficient large-scale 3D reconstruction and streaming for immersive remote collaboration based on consumer hardware is illustrated in Fig. 3 and the tasks of the involved components are shown in Fig. 4. RGB-D data acquired with commodity 3D depth sensors as present in a growing number of smartphones or the Kinect device are sent to the reconstruction client, where the 3D model of the scene is updated in real time and transmitted to the server. The server manages a copy of the reconstructed model, a corresponding, novel, bandwidth-optimized voxel block representation, and the further communication with connected

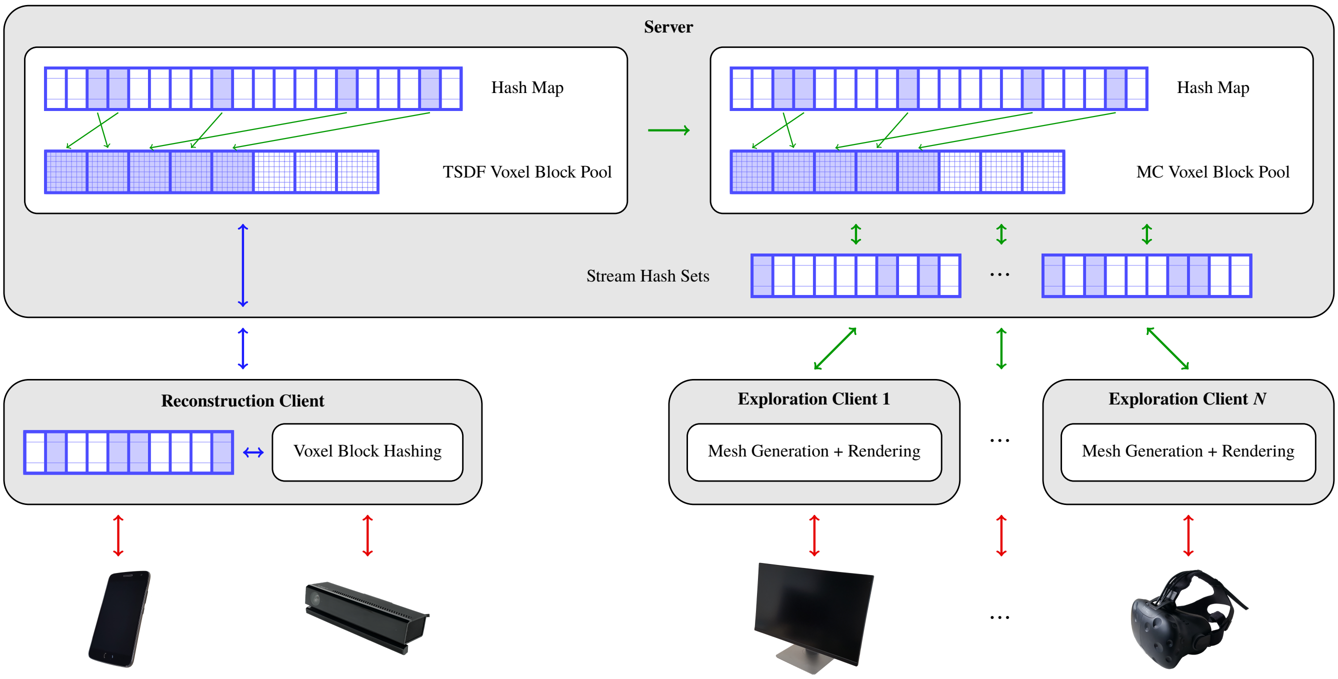


Fig. 3. Our novel 3D reconstruction and streaming framework for multi-client remote collaboration. RGB-D images acquired by consumer cameras, e.g. smartphones or the Kinect device, are streamed to the reconstruction client (red arrows) which updates the virtual model and transfers it to the server (blue arrows). The server converts the received data to a novel bandwidth-optimized representation based on Marching Cubes (MC) indices and manages a set of updated blocks that are queued for streaming for each connected exploration client. By design, our system supports an arbitrary number of exploration clients that can independently request the currently relevant updated parts of the model (green arrows) and integrate them into their locally generated mesh from which images are rendered in real-time and displayed on devices such as VR headsets or screens. For an immersive lag-free experience, the computational load during streaming is distributed using our novel hash map and set data structures. Red arrows are used to represent the image streaming, while blue and green arrows are used to represent the streaming of TSDF and MC voxel blocks.

exploration clients. Finally, at the exploration client, the transmitted scene parts are triangulated to update the locally generated mesh which can be immersively explored i.e. with VR devices. Clients can connect at any time before or after the capturing process has started. In the following sections, we provide more detailed descriptions of the individual components of our framework, i.e. the reconstruction client, the server, and the exploration client, which is followed by an in-depth discussion of the novel data structure (see Sect. 5). Additional implementation details for each component are provided in the supplemental material.

4.1 Reconstruction Client

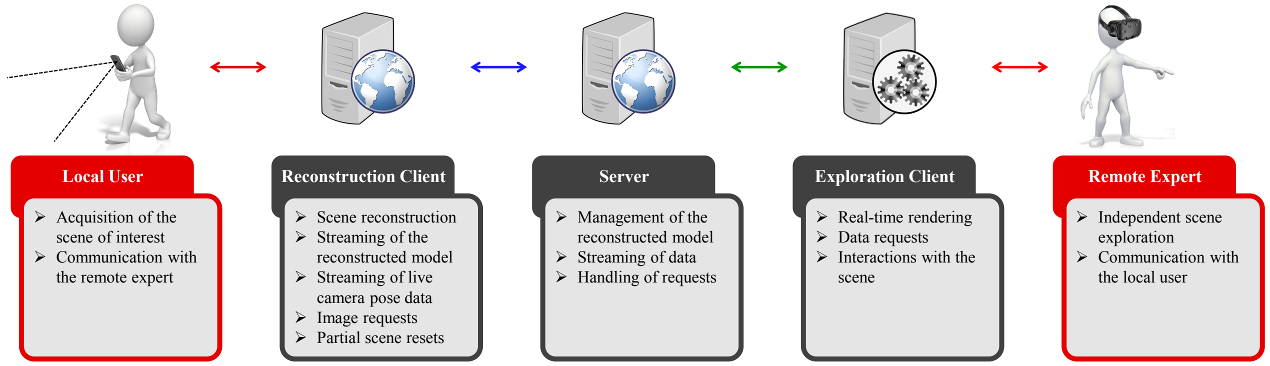
The reconstruction client receives a stream of RGB-D images acquired by a user and is responsible for the reconstruction and streaming of the virtual model. We use voxel block hashing [22, 39] to reconstruct a virtual 3D model from the image data. Since the bandwidth is limited, the as-efficient-as-possible data handling during reconstruction is of great importance. For this purpose, we consider only voxel blocks that have already been fully reconstructed and for which no further immediate updates have to be considered, i.e. blocks that are not visible in the current sensor's view anymore and have been streamed out to CPU memory [35]. In contrast, transmitting blocks that are being still actively reconstructed and, thus, will change over time which results in an undesirable visualization experience for exploration clients. Furthermore, continuously transmitting these individual blocks during the reconstruction process results in extremely increasing bandwidth requirements which make this approach infeasible to real-world scenarios. In contrast to Mossel and Kröter [35], we concurrently insert the streamed-out voxel blocks into a hash set which allows us to control the amount of blocks per package that are streamed and avoids lags by distributing the work across multiple frames similar to the transfer buffer approach of the InfiniTAM system [22]. To mitigate the delay caused by transmitting only fully reconstructed parts of the scene, we add the currently visible blocks at the very end of the acquisition process as well as when the user stops moving during capturing or the hardware including the network connection are powerful enough to stream the complete amount of queued entries. In particular, we check whether the exponential moving average (EMA) of the stream set size over a period of $\tau = 5$ seconds [53] is below a given threshold and the last such prefetching operation is at least 5 seconds ago. The EMA is updated as

$$EMA_{\tau}^{(t_{n+1})} = u\,EMA_{\tau}^{(t_n)} + (v-u)\,s_n + (1-u)\,s_{n+1} \tag{1}$$

with

$$u = e^{-a}, \quad v = \frac{1-u}{a}, \quad a = \frac{t_{n+1}-t_n}{\tau}. \tag{2}$$

This ensures that the delayed but complete model is available to the server and the exploration clients at all times. After fetching a subset of stream set (via concurrent removal) and the respective voxel data from the model, we compress them using lossless compression [5] and send them to the server. In addition to the pure voxel data, the reconstruction client and the exploration clients send their camera intrinsics and current camera pose to the server where they are forwarded to each connected exploration client to enable interactive collaboration. Furthermore, requests for high-resolution textures on the model by the exploration clients, required e.g. for reading text or measurement instruments, are handled by transmitting the sensor's current RGB image to the reconstruction client where it is forwarded to the server and the exploration clients. To make our framework also capable of handling quasi-static scenes, where the scene is allowed to change between two discrete timestamps, as e.g. occurring when an instrument cabinet has to be opened before being able to read the instruments, our framework also comprises a reset function that allows the exploration client to request scene updates for selected regions. This can be achieved by deleting the reconstructed parts of the virtual model that are currently visible and propagating the list of these blocks to the server.



Local User
Acquisition of the scene of interest
Communication with the remote expert
Reconstruction Client
Scene reconstruction
Streaming of the reconstructed model
Streaming of live camera pose data
Image requests
Partial scene resets
Server
Management of the reconstructed model
Streaming of data
Handling of requests
Exploration Client
Real-time rendering
Data requests
Interactions with the scene
Remote Expert
Independent scene exploration
Communication with the local user

Fig. 4. Components of our framework and their respective tasks. Images are partially provided by PresenterMedia [42].

4.2 Server

The server component is responsible for managing the global voxel block model and the list of queued blocks for each connected exploration client. Furthermore, it converts incoming TSDF voxel blocks into our novel MC voxel block representation. Finally, it forwards messages between clients and distributes camera and client pose data for an improved immersion and client interaction.

In order to reduce the computational burden and infrastructural requirements regarding network bandwidth, the streamed data should be as compact as possible while being efficiently to process. Instead of streaming the model in the original TSDF voxel block representation of the voxel block hashing technique [35] to the exploration clients, we compute and transmit a bandwidth-optimized representation based on Marching Cubes [30]. Thus, a TSDF voxel (12 bytes), composed of a truncated signed distance field (TSDF) value (4 bytes), a fusion weight (4 bytes), and a color (3 bytes + 1 byte alignment), is reduced to a MC voxel, i.e. a Marching Cubes index (1 byte), and a color value (3 bytes). Furthermore, we cut off those voxel indices i and colors c where no triangles will be created, i.e. for

$$\mathscr{S}^c = \{(i,c) \mid i = 0 \vee i = 255\}\,, \tag{3}$$

by setting the values i and c to zero. While omitting the interpolation weights, resulting in lossy compression, might seem drastic in terms of reconstruction quality, we show that the achieved improvement regarding compression ratio and network bandwidth requirement outweigh the slight loss of accuracy in the reconstruction (see Sect. 6). Compared to a binary representation of the geometry that would lead to the same quality and a similar compression ratio, our MC index structure directly encodes the triangle data and enables the independent and parallel processing at the remote site by removing neighborhood dependencies.

Incoming data sent by the reconstruction client are first concurrently integrated into the TSDF voxel block model and then used to update the corresponding blocks and their seven neighbors in negative direction in the MC voxel block representation. Updating the neighbors is crucial to avoid cuts in the mesh due to outdated and inconsistent MC indices. To avoid branch divergence and inefficient handling of special cases, we recompute the whole blocks instead of solely recomputing the changed parts. The list of updated MC voxel blocks is then concurrently inserted to each exploration client's stream hash set. Maintaining such a set for each connected client not only enables advanced streaming strategies required for a lag-free viewing experience (see Sect. 4.3). It also allows them to reconnect at any point in time, e.g. after network outages, and still explore the entire model since their stream sets are initially filled with the complete list of voxel blocks via concurrent insertion. After selecting all relevant blocks, a random subset of at most the request size limit is extracted via concurrent removal and the corresponding voxel data are retrieved, compressed [5] and sent to the exploration client.

4.3 Exploration Client

The exploration client's tasks comprise generating surface geometry from the transmitted compact representation in terms of MC indices, updating the current version of the reconstructed model at the remote site, and the respective rendering of the model in real-time. Therefore, exploration clients are allowed to request reconstructed voxel blocks according to the order of their generation during reconstruction, depending on whether they are visible in the current view of the client, or in a random order which is particularly useful in the case when the currently visible parts of the model are already complete, and thus, other parts of the scene can be prefetched. Since the exploration client controls the request rate and size, a lag-free viewing experience is achieved by adapting these parameters depending on the client's hardware resources.

The received MC voxel blocks are decompressed in a dedicated thread, and the block data is passed to a set of reconstruction threads which generate the scene geometry from the MC indices and colors of the voxels. We reduce the number of draw calls to the graphics API by merging 15^3 voxel blocks into a mesh block instead of rendering each voxel block separately [35]. To reduce the number of primitives rendered each frame, we compute three level of details (LoDs) from the triangle mesh, where one voxel, eight voxels or 64 voxels respectively are represented by a point and the point colors are averaged over the voxels. During the rendering pass, all visible mesh blocks are rendered, while their LoD is chosen according to the distance from the camera. We refer to the supplemental material for more details.

To allow a better interaction between the involved clients, each exploration client additionally sends its own pose to the server, which distributes it to other exploration clients, so that each user can observe the poses and movements of other exploration clients within the scene. Analogously, the current pose of the reconstruction client is visualized in terms of the respectively positioned and oriented camera frustum. Furthermore, users can interactively explore the reconstructed environment beyond pure navigation by measuring 3D distances between interactively selected scene points. For the purpose of depicting structures below the resolution of the voxel hashing pipeline as e.g. required for reading measurement instruments or texts, the exploration client can send requests to the server upon which the RGB image currently captured by the sensor is directly projected onto the respective scene part and additionally visualized on a virtual measurement display.

5 Hash Map and Set Data Structures

For the purpose of large-scale 3D reconstruction and streaming to an arbitrary number of remote exploration clients, we developed a thread-safe GPU hash data structure allowing fast and simple management including dynamic concurrent insertion, removal and retrieval of millions of entries with strong success guarantees. In comparison to pure 3D reconstruction, maintaining consistency in multi-client telepresence is much more challenging since streaming data between clients requires that updates are not lost e.g. due to synchronization failures. Whereas previous approaches either allow failures [14, 22, 39] or do not ensure key uniqueness [2, 25], our robust hash data structure is not limited in this regard and represents the key to realize our real-time remote collaboration system. A detailed evaluation in terms of design choices

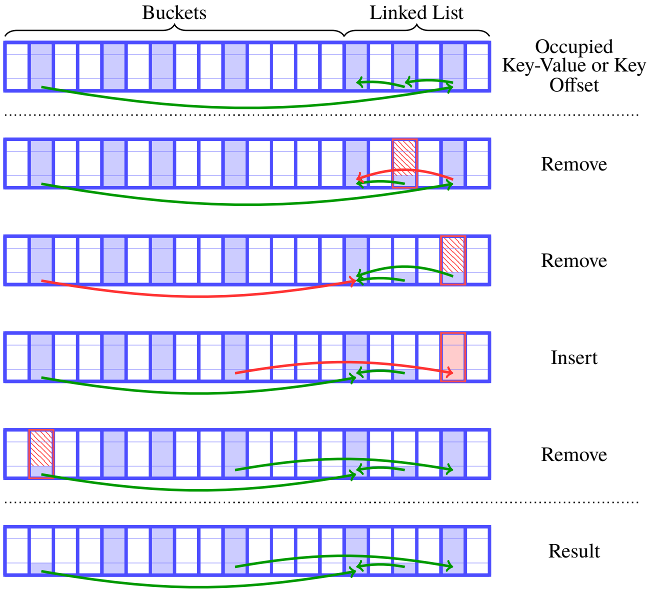


Fig. 5. Illustration of thread-safe hash map/set modifications on the GPU by maintaining the proposed invariant. The importance of thread safety has its origin in the guarantees for successful concurrent retrieval, insertion and removal while preserving key uniqueness. This figure depicts one possible order for the operations to resolve the requested task when processing four operations in parallel. In the resulting structure, dead links and empty buckets might occur which, however, are not problematic and automatically cleaned up during further operations.

and runtime performance can be found in the supplemental material.

General Design Our streaming pipeline is built upon two different hash data structures. The server and the individual client components use an internal map structure, that stores unique keys and maps a value to each of them, whereas the server-client streaming protocol relies on a set structure, which only considers the keys. Thus, the major difference lies in the kind of stored data whereas the proposed algorithm for retrieval, insertion and removal is shared among them. We built upon the single-entry data structure by Kähler et al. [22] which stores the values, i.e. key-value pairs for the map structure (voxel block hashing and server model) and keys for the set (streaming states, see Fig. 3) into a linear array. Collisions are resolved through linked lists using per-entry offsets to the next elements and a stack structure that maintains the set of available linked list entries. Voxel block hashing based reconstruction approaches rely on the high camera frame rate to clean up block allocation failures in subsequent frames [7, 22, 23, 39] and, therefore, reduce synchronization to a minimum. In contrast, failures in our telepresence system result in data loss during data transmission which cannot be recovered. Thus, we need additional indicators to determine whether an entry is occupied and locks for synchronization to handle cases where several threads attempt to modify the same entry simultaneously. Furthermore, we maintain a strong invariant which is required to achieve correct concurrency on the thread-level: *At any time, the entry positions and the links to colliding values are preserved.* Fig. 5 demonstrates mixed insertion and removal operations on our thread-safe hash data structure. Detailed descriptions and implementation details of the hash and stack data structures as well as further design remarks are provided in the supplemental material.

Retrieval Since our proposed invariant ensures that entry positions are not allowed to change, finding an element in the hash map or set can be safely implemented as a read-only operation. First, the bucket b of a given key value is computed according to the underlying hashing function. In case of spatial hashing, this function could be defined as

$$b = (x \cdot p_1 \oplus y \cdot p_2 \oplus z \cdot p_3) \bmod n \tag{4}$$

where (x, y, z) are the voxel block coordinates, $p_1 = 73856093, p_2 = 19349669, p_3 = 83492791$ represent prime numbers, and n denotes the number of buckets [22, 39]. We check whether the entry is occupied and its key matches the query. If both conditions are met, we found the key and return the current position. Otherwise, we traverse the linked list through the offsets and check each entry in a similar manner.

Insertion For successful concurrent insertion, the modification of an entry by several threads needs to be handled while avoiding deadlocks. We handle the latter problem by by looping over a non-blocking insertion function, which is allowed to fail, until the value is found in the data structure. In the non-blocking version, we first check if the value is already inserted (by performing retrieval). If the entry is not found, there are two possible scenarios: The value can be inserted at the bucket (if this entry is not occupied) or at the end of the bucket's linked list. In both cases, other threads might attempt to also modify the entry at the same time. This not only requires locking (which might fail to prevent deadlocks), but also a second occupancy check. If both the lock is successfully acquired and the entry is still free, the value is stored and the entry is marked as occupied and unlocked. In case the bucket was initially occupied (second scenario), we first find the end of the linked list by traversing the offsets and lock that entry. Afterwards, we extract a new linked list position from the stack, store the value there, set the occupancy flag and reset its offset to zero. Note that the offset is intentionally not reset in the removal operation to avoid a race condition (see the section below for details). Finally, the offset to the new linked list entry is stored and the acquired lock is released.

Removal Removing elements as required when selecting voxel blocks for client-server streaming, is similar to insertion and also involves double checking during lock acquisition as well as looping over a non-blocking version. Again, there are two possible scenarios: The entry may be located at the bucket or inside the linked list. In the former case, we try to acquire the lock and then reset the value and mark the entry as unoccupied. In contrast to the approach by Nießner et al. [39], the first linked list entry is not moved to the bucket to preserve our invariant. Threads that try to erase this value might, otherwise, fail to find it. We evaluated the impact of this change and observed that runtime performance was not affected. If the value is inside the linked list (second scenario), we first find the previous entry and lock both entries. Afterwards, the current entry is reset and marked as unoccupied, the offset of the previous entry is updated, and both locks are finally released. As mentioned earlier, the offset is kept to avoid a race condition where other threads concurrently performing direct or indirect retrieval (inside insertion and removal) might not be able to access the remainder of the linked list which would lead to failures in all three operations. Thus, we avoid the need for additional synchronization in the retrieval operation by delaying this step to the insertion operation.

6 Evaluation

After providing implementation details, we perform an analysis regarding bandwidth requirements and the visual quality of our compact scene representation. This is accompanied by the description of the usage of our framework in a live remote collaboration scenario as well as a discussion of the respective limitations.

6.1 Implementation

We implemented our framework using up to four desktop computers taking the roles of one reconstruction client, one server, and two exploration clients. Each of the computers has been equipped with an Intel Core i7-4930K CPU and 32GB RAM. Furthermore, three of them have been equipped with a NVIDIA GTX 1080 GPU with 8GB VRAM, whereas the fourth computer made use of a NVIDIA GTX TITAN X GPU with 12GB VRAM. For acquisition, we tested two different RGB-D sensors by using the Microsoft Kinect v2, which delivered data with a resolution of 512 × 424 pixels at 30Hz, and by using an ASUS Zenfone AR, which captured RGB-D data with a resolution of 224 × 172 pixels at 10Hz. Although the ASUS device is, in principle, capable of performing measurements at frame rates of 5-15Hz, we used 10Hz as a compromise between data completeness and speed. Each of the

Table 2. Bandwidth measurements of our system for various scenes. We compared mean (and maximum) bandwidths of our optimized MC voxel structure with 128-1024 blocks/request and 100Hz request rate to the standard TSDF representation with 512 blocks/request and unlimited rate. Across all scenes, our optimized representation saved more than 90% of the bandwidth and scales linearly with the package size.

Dataset	Voxel Size [mm]	Bandwidth [MBit/s]					Model Size [# Voxel Blocks]
		MC 128	MC 256	MC 512	MC 1024	TSDF 512	
heating_room	5	4.5 (8.0)	8.8 (12.3)	17.5 (30.9)	32.7 (71.3)	561.5 (938.8)	897 $\times 10^3$
pool	5	4.6 (7.1)	9.0 (14.0)	17.8 (29.7)	29.3 (54.5)	489.3 (937.0)	637 $\times 10^3$
fr1/desk2	5	8.1 (11.6)	16.2 (23.8)	32.6 (46.8)	61.0 (95.0)	764.0 (938.6)	134 $\times 10^3$
fr1/room	5	12.3 (23.6)	16.4 (23.6)	32.1 (42.2)	57.6 (87.9)	739.7 (938.0)	467 $\times 10^3$
heating_room	10	5.1 (7.6)	9.2 (14.4)	14.6 (27.8)	20.2 (63.7)	216.8 (937.1)	147 $\times 10^3$
pool	10	5.6 (8.5)	9.9 (16.0)	13.6 (27.2)	16.9 (52.3)	176.3 (937.0)	104 $\times 10^3$
fr1/desk2	10	8.7 (11.2)	14.3 (21.8)	19.6 (39.2)	24.4 (71.3)	170.1 (436.4)	23 $\times 10^3$
fr1/room	10	9.2 (12.5)	15.7 (23.5)	22.9 (46.1)	28.5 (88.8)	207.8 (936.6)	86 $\times 10^3$

Table 3. Time measurements of our system for various scenes. We compared the time to stream the whole model represented by our optimized MC voxel structure with 128-1024 blocks/request and 100Hz request rate to the standard TSDF representation with 512 blocks/request and unlimited rate. The reconstruction speed is given by TSDF 512 and serves as a lower bound. For a voxel resolution of 5mm, a package size of 512 voxel blocks results in the best trade-off between required bandwidth and total streaming time. Increasing the size leads to slightly better results with less latency, but substantially higher bandwidths. For a resolution of 10mm, the optimal streaming time is reached with even smaller package sizes.

Dataset	Voxel Size [mm]	Time [min]					Model Size [# Voxel Blocks]
		MC 128	MC 256	MC 512	MC 1024	TSDF 512	
heating_room	5	4:06	3:08	2:40	2:32	2:31	897 $\times 10^3$
pool	5	2:14	1:32	1:12	1:09	1:08	637 $\times 10^3$
fr1/desk2	5	0:39	0:31	0:27	0:24	0:22	134 $\times 10^3$
fr1/room	5	1:46	1:14	1:01	0:57	0:56	467 $\times 10^3$
heating_room	10	1:49	1:44	1:44	1:44	1:44	147 $\times 10^3$
pool	10	0:54	0:50	0:50	0:50	0:50	104 $\times 10^3$
fr1/desk2	10	0:21	0:19	0:19	0:19	0:18	23 $\times 10^3$
fr1/room	10	0:46	0:42	0:41	0:41	0:41	86 $\times 10^3$

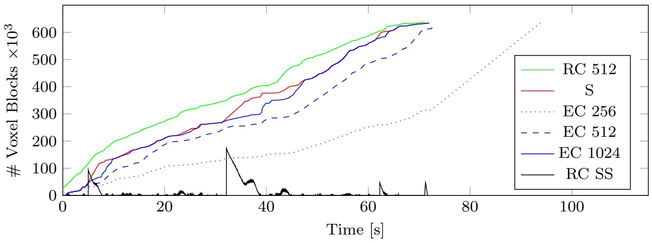


Fig. 6. Streaming progress over time for the *pool* dataset. Larger package sizes reduce the total transmission time of the virtual model to the exploration client (EC). To save bandwidth, only fully reconstructed blocks are streamed from the reconstruction client (RC) to the server (S) causing a noticeable delay, which becomes smaller when our prefetching queues the currently visible scene parts to the RC's stream set (RC SS).

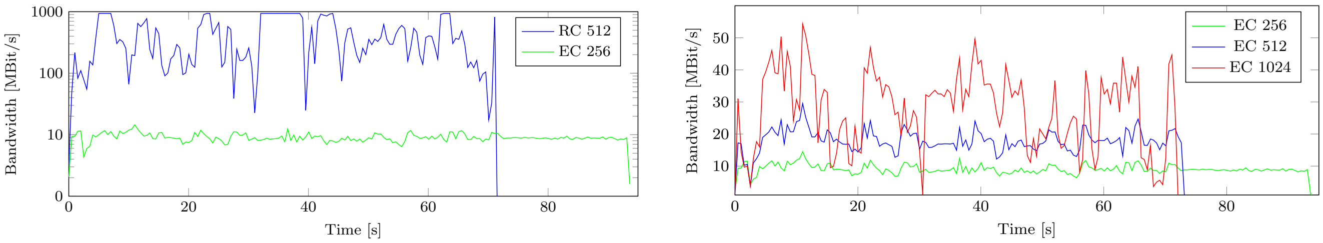
exploration client users was equipped with an HTC Vive HMD with a native resolution of 1080 × 1200 pixels per eye whereas the recommended rendering resolution (reported by the VR driver) is 1512 × 1680 pixels per eye, leading to a total resolution of 3024 × 1680 pixels. Please note that the higher recommended resolution (in comparison to the display resolution) originates from the lens distortion applied by the VR system. All computers were connected via a local network.

6.2 Bandwidth and Latency Analysis

In the following, we provide a detailed quantitative evaluation of the bandwidth requirements of our novel collaboration system. For the purpose of comparison, we recorded two datasets *heating_room* and *pool* (see supplemental material) with the Kinect v2, and also used two further publicly available standard datasets that were captured with the Kinect v1 [45]. Throughout the experiment, we loaded a dataset and performed the reconstruction on the computer equipped with the NVIDIA GTX TITAN X. The model is then streamed to the server (second computer) and further to a benchmark client (third computer). Compared to the exploration client, the benchmark client is started simultaneously to the reconstruction client, requests voxel blocks with a fixed predefined frame rate of 100Hz, and directly discards the received data to avoid overheads. Using this setup, we measured the mean and maximum bandwidth required for streaming the TSDF voxel block model from the reconstruction client to the server and the MC voxel block model from the server to the benchmark client. Furthermore, we also measured the time until the model has been completely streamed to the benchmark client. For the voxel block hashing pipeline, we used 5mm and 10mm for the voxel size, 60mm for the truncation region and hash maps with 2^{20} and 2^{22} buckets as well as excess list sizes matching the respective active GPU and passive CPU voxel block pool sizes of 2^{19} and 2^{20} blocks. The server and reconstruction client used the passive parameter set for their hash maps and sets. The results of our experiment are shown in Table 2 and Table 3. A further evaluation regarding the server scalability is provided in the supplemental material.

Across all scenes and voxel sizes, the measured mean and maximum bandwidths for our novel MC voxel structure scale linearly with the package size and are over one order of magnitude smaller compared to the standard TSDF voxel representation. We measured higher bandwidths at 10mm voxel size than at 5mm for package sizes of 128 and 256 blocks. Our stream hash set automatically avoids duplicates, which saves bandwidth in case the system works at its limits and can be considered as an adaptive streaming. At 10mm this triggers substantially less and thus, more updates are sent to the server and exploration clients. We also observed by a factor of two larger bandwidths for the datasets captured with the Kinect v1 in comparison to the ones recorded by us with the Kinect v2. This is mainly caused by the lower reliability of the RGB-D data which contains more sensor noise as well as holes, which, in turn, results in a larger number of allocated voxel blocks that need to be streamed. Furthermore, the faster motion induces an increased motion blur within the images, and thus leads to larger misalignments in the reconstructed model as well as even more block allocations. However, this problem is solely related to the reconstruction pipeline and does not affect the scalability of our collaboration system.

The overall system latency is determined by the duration until newly



(a) Bandwidth requirements between reconstruction client (RC) and server (S) with 512 blocks/request and an exploration client (EC) with 256 blocks/request. As both RC and S are within the same network (i.e. in the cloud) in the proposed system architecture, the shown bandwidth requirements are still acceptable.

(b) Bandwidth requirements between server (S) and exploration client (EC) with package sizes of 256, 512, and 1024 blocks/request.

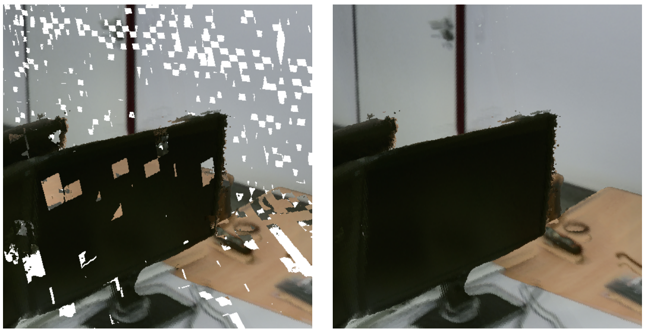
Fig. 7. Bandwidth measurements of our system over time for the *pool* dataset.

seen parts of the scene are queued for transmission, i.e. until they are streamed out to CPU memory, the latency of the network, and the package size of the exploration client's requests. Since the whole system runs in real-time, i.e. data are processed in the order of tens of milliseconds, the runtime latency within the individual components has a negligible impact on the total latency of the system. In order to evaluate the bandwidth requirements and the overall latency, we performed further measurements as depicted in Fig. 7 and Fig. 6. Whereas the bandwidth for transmitting the TSDF voxel block representation has a high variance and ranges up to our network's limit of 1Gbit/s, our bandwidth optimized representation has not only lower requirements, i.e. a reduction by more than 90%, but also a significantly lower variance. For a package size of 256 blocks, the model is only slowly streamed to the exploration client which results in a significant delay until the complete model has been transmitted. Larger sizes such as 512 blocks affect both the mean bandwidth and the variance while further increases primarily affect the variance since less blocks than the package size need to be streamed (see Fig. 7). This effect also becomes apparent in Fig. 6 where lower package sizes lead to a smooth streaming and larger delays whereas higher values reduce the latency. Furthermore, the delay between the reconstruction client and the server in the order of seconds is directly related to our choice of only transmitting blocks that have been streamed out to save bandwidth. Note that directly streaming the actively reconstructed voxel blocks is infeasible due to extremely increasing bandwidth requirements (see Section 4.1). Once our automatic streaming of the visible parts triggers, which can be seen in the rapid increases of the RC's stream set (RC SS), the gap between the current model at the reconstruction client and the streamed copy at the server becomes smaller. Since the visible blocks are streamed in an arbitrary order, this results in lots of updates for already existing neighboring MC voxel blocks at the server site that need to be streamed to the exploration client. Therefore, the exploration client's model grows slower than the server's model but this gap is closed shortly after the server received all visible blocks. Note that the effects of this prefetching approach can be also seen in the reconstruction client's bandwidth requirements, where high values are typically observed when this mechanism is triggered.

In comparison to per-frame streaming [35], we transmit data per block which allows the recovery from network outages as well as advanced streaming strategies controlled by the remote user. Therefore, depending on the possibly very high number of eligible blocks from streaming, e.g. all visible blocks after re-connection, scene updates may appear unordered and patch-by-patch which can affect the subjective latency (see the supplemental video). However, due to the controllable strategies, the objective latency until these visible data are fully transmitted is much smaller than for inflexible frame-based approaches.

6.3 Scene Model Completeness and Visual Quality

In addition to the bandwidth analysis, we have also evaluated the model completeness during transmission for our novel hash map data structure in comparison to previous techniques that allow failures [39]. Thus, we measured the model size in terms of voxel blocks at the reconstruction client, where the streaming starts, and at the exploration client, where the data is finally transmitted to. To reduce side effects caused by distributing the computational load, we have chosen a package size of 1024 blocks (see Table 3). Whereas previous GPU hashing techniques work well for 3D reconstruction and failures can be cleaned up in subsequent frames, they are not suitable for large-scale collaboration scenarios where blocks are often sent only once to save bandwidth. Insertion and removal failures will, hence, lead to holes in the reconstruction that cannot be repaired in the future (see Fig. 8).



(a) Hash Map by Nießner et al. [39].

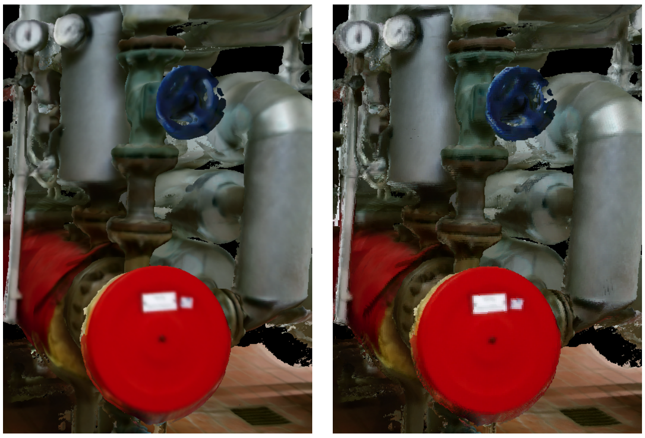
(b) Our Hash Map Data Structure.

Fig. 8. Visual comparison of model completeness for the *pool* dataset: While previous hash maps allow failures, our hash data structure ensures hole-free reconstructions during transmission to an exploration client.

We also provide a qualitative visual comparison of our bandwidth-optimized scene representation based on Marching Cubes indices. In order to reduce the bandwidth requirements by over 90%, we omitted the interpolation of vertex positions and colors. Fig. 9 shows a comparison between our approximation and the interpolated mesh, where both representations have been reconstructed using a voxel resolution of 5mm. While the interpolated model has a smooth appearance, the quality of our approximation is slightly lower at edges but, otherwise, resembles the overall visual quality quite well. However, for small highly textured objects, staircase artifacts become visible and lead to worse reconstruction results (see Fig. 10). Note that our system allows compensating this issue by using our projective texture mapping approach to enable higher resolution information on demand.

6.4 Live Remote Collaboration

To verify the usability of our framework, we conducted a live remote collaboration experiment where a local user and two remotely connected users collaboratively inspect the local user's environment supported by audio-communication (i.e. via Voice over IP (VoIP)). For this experiment, we selected people who were unfamiliar to our framework and received a briefing regarding the controls. Furthermore, these user have never been in the respective room before.



(a) With Color Interpolation. (b) Without Color Interpolation.

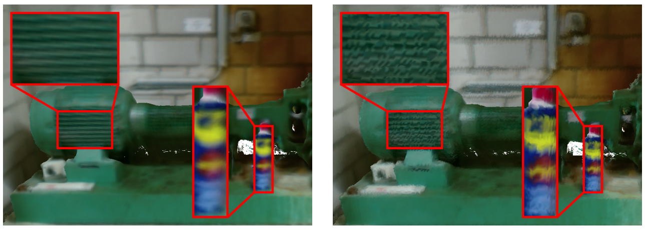
Fig. 9. Visual comparison of our scene encoding for the *heating_room* dataset: Compared to standard mesh generation techniques that use linear interpolation, our scene encoding achieves a similar quality without interpolation in real-world scenes.

While one person took the role of a local user operating the acquisition device, two different remotely connected exploration clients provide support regarding maintenance and safety. The exploration clients can interactively inspect the acquired scene, i.e. the maintenance expert guides the person operating the acquisition device to allow the observation of measurement instruments. By allowing scene resets, where parts of the scene can be updated on demand, our system allows certain scene manipulations such as opening the door to a switch board that has to be checked by the maintenance expert. Furthermore, the scene model can be visualized at higher texture resolution based on the transmission of the live-captured RGB image upon request and its usage in a separate virtual 2D display or directly on the scene geometry. This allows checking instruments or even reading text (see supplemental material for further details and evaluation). Measurements performed based on the controllers belonging to the HMD devices are of sufficient accuracy to allow detecting safety issues or select respective components for replacement. The interaction flow of this experiment is also showcased in the supplemental video. In addition to the Kinect v2, we also used an ASUS Zenfone AR (224 × 172 pixels, up to 15Hz) for RGB-D acquisition. However, the limited resolution and frame rate affect the reconstruction quality obtained with the smartphone.

Furthermore, the users testing our framework particularly liked the options to reset certain scene parts to get an updated scene model as well as the possibility of interacting with the scene by performing measurements and inspecting details like instrument values. After network outages or wanted disconnections from the collaboration process, the capability of re-connecting to re-explore the in-the-meantime reconstructed parts of the scene was also highly appreciated and improved the overall experience significantly. In fact, they reported a good spatial understanding of the environment.

6.5 Limitations

Despite allowing an immersive live collaboration between an arbitrary number of clients, our system still faces some limitations. In particular, the acquisition and reconstruction of a scene with a RGB-D camera may be challenging for unexperienced users, who tend to move and turn relatively fast resulting in high angular and linear velocities as well as potential motion blur. As a consequence, the reconstruction is more susceptible to misalignments. Whereas loop-closure techniques [7] compensate this issue, their uncontrollable update scheme during loop closing would cause nearly the entire model to be queued for streaming. This would impose much higher bandwidth requirements to the client connections and prohibit remote collaboration over the Internet. Submap approaches [21] avoid this problem, but issues such as the submap jittering caused by progressive relocalization during the capturing process have to be handled carefully in order to preserve an acceptable VR experience and require a respective evaluation in the scope of a comprehensive user study. Furthermore, we stream the virtual model in the TSDF voxel representation between the reconstruction client and the server which requires both to be in a local network. However, the increasing thrust in cloud services could fill this gap. While we believe that the usability of our novel system significantly benefits from mobile devices with built-in depth cameras, the current quality and especially the frame rate of the provided RGB-D data is inferior compared to the Kinect family resulting in low-quality reconstructions.



(a) With Color Interpolation. (b) With Color Interpolation.

Fig. 10. Challenging cases: For highly textured objects and sharp edges with high contrasts, our approximation introduces small artifacts.

7 Conclusion

We presented a novel large-scale 3D reconstruction and streaming framework for immersive multi-client live telepresence that is especially suited for remote collaboration and consulting scenarios. Our framework takes RGB-D inputs acquired by a local user with commodity hardware such as smartphones or the Kinect device from which a 3D model is updated in real-time. This model is streamed to the server which further manages and controls the streaming process to the, theoretically, arbitrary number of connected remote exploration clients. As such as system needs to access and process the data in highly asynchronous manner, we have built our framework upon – to the best of our knowledge – the first thread-safe GPU hash map data structure that guarantees successful concurrent insertion, retrieval and removal on a thread level while preserving key uniqueness required by current voxel block hashing techniques. Efficient streaming is achieved by transmitting a novel, compact representation in terms of Marching Cubes indices. In addition, the inherently limited resolution of voxel-based scene representations can be overcome with a lightweight projective texture mapping approach which enables the visualization textures at the resolution of the depth sensor of the input device. As demonstrated by a variety of qualitative experiments, our framework is efficient regarding bandwidth requirements, and allows a high degree of immersion into the live captured environments.

Acknowledgments

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] D. A. F. Alcantara. *Efficient Hash Tables on the GPU*. PhD thesis, University of California at Davis, 2011.

[2] S. Ashkiani, M. Farach-Colton, and J. D. Owens. A Dynamic Hash Table for the GPU. In *IEEE Int. Parallel and Distributed Processing Symposium*, pp. 419–429, 2018.

[3] F. C. Botelho, R. Pagh, and N. Ziviani. Practical Perfect Hashing in Nearly Optimal Space. *Inf. Syst.*, 38(1):108–131, 2013.

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

[5] Y. Collet and C. Turner. Smaller and faster data compression with Zstandard. https://code.fb.com/core-data/

smaller-and-faster-data-compression-with-zstandard/, 2016. Accessed: 2019-01-29.

[6] X. Corbillon, G. Simon, A. Devlic, and J. Chakareski. Viewport-adaptive navigable 360-degree video delivery. In *2017 IEEE Int. Conf. on Communications*, pp. 1–7, 2017.

[7] 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(3):24, 2017.

[8] M. Dou et al. Fusion4D: Real-time Performance Capture of Challenging Scenes. *ACM Trans. Graph.*, 35(4):114:1–114:13, 2016.

[9] A. J. Fairchild, S. P. Campion, A. S. García, R. Wolff, T. Fernando, and D. J. Roberts. A Mixed Reality Telepresence System for Collaborative Space Operation. *IEEE Trans. on Circuits and Systems for Video Technology*, 27(4):814–827, 2016.

[10] C.-L. Fan, J. Lee, W.-C. Lo, C.-Y. Huang, K.-T. Chen, and C.-H. Hsu. Fixation Prediction for 360° Video Streaming in Head-Mounted Virtual Reality. In *Proc. of the 27th Workshop on Network and Operating Systems Support for Digital Audio and Video*, pp. 67–72, 2017.

[11] G. Fontaine. The Experience of a Sense of Presence in Intercultural and Int. Encounters. *Presence: Teleoper. Virtual Environ.*, 1(4):482–490, 1992.

[12] H. Fuchs, G. Bishop, K. Arthur, L. McMillan, R. Bajcsy, S. Lee, H. Farid, and T. Kanade. Virtual Space Teleconferencing Using a Sea of Cameras. In *Proc. of the Int. Conf. on Medical Robotics and Computer Assisted Surgery*, pp. 161 – 167, 1994.

[13] H. Fuchs, A. State, and J. Bazin. Immersive 3D Telepresence. *Computer*, 47(7):46–52, 2014.

[14] I. García, S. Lefebvre, S. Hornus, and A. Lasram. Coherent Parallel Hashing. *ACM Trans. Graph.*, 30(6):161:1–161:8, 2011.

[15] S. Golodetz, T. Cavallari, N. A. Lord, V. A. Prisacariu, D. W. Murray, and P. H. S. Torr. Collaborative Large-Scale Dense 3D Reconstruction with Online Inter-Agent Pose Optimisation. *IEEE Trans. on Visualization and Computer Graphics*, 24(11):2895–2905, Nov 2018.

[16] R. M. Held and N. I. Durlach. Telepresence. *Presence: Teleoper. Virtual Environ.*, 1(1):109–112, 1992.

[17] P. Henry, D. Fox, A. Bhowmik, and R. Mongia. Patch Volumes: Segmentation-Based Consistent Mapping with RGB-D Cameras. In *Int. Conf. on 3D Vision*, 2013.

[18] M. Hosseini and V. Swaminathan. Adaptive 360 VR Video Streaming: Divide and Conquer. *IEEE Int. Symp. on Multimedia*, pp. 107–110, 2016.

[19] S. Izadi et al. KinectFusion: Real-time 3D Reconstruction and Interaction Using a Moving Depth Camera. In *Proc. of the ACM Symp. on User Interface Software and Technology*, pp. 559–568, 2011.

[20] B. Jones et al. RoomAlive: Magical Experiences Enabled by Scalable, Adaptive Projector-camera Units. In *Proc. of the Annual Symp. on User Interface Software and Technology*, pp. 637–644, 2014.

[21] 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*, pp. 500–516, 2016.

[22] 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. *IEEE Trans. on Visualization and Computer Graphics*, 21(11):1241–1250, 2015.

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

[24] T. Kanade, P. Rander, and P. J. Narayanan. Virtualized reality: constructing virtual worlds from real scenes. *IEEE MultiMedia*, 4(1):34–47, 1997.

[25] F. Khorasani, M. E. Belviranli, R. Gupta, and L. N. Bhuyan. Stadium Hashing: Scalable and Flexible Hashing on GPUs. In *Proc. of the Int. Conf. on Parallel Architecture and Compilation*, pp. 63–74, 2015.

[26] A. Kunert, A. Kulik, S. Beck, and B. Froehlich. Photoportals: Shared References in Space and Time. In *Proc. of the 17th ACM Conf. on Computer Supported Cooperative Work & Social Computing*, pp. 1388–1399, 2014.

[27] G. Kurillo, R. Bajcsy, K. Nahrsted, and O. Kreylos. Immersive 3D Environment for Remote Collaboration and Training of Physical Activities. In *IEEE Virtual Reality Conference*, pp. 269–270, 2008.

[28] S. Lefebvre and H. Hoppe. Perfect Spatial Hashing. *ACM Trans. Graph.*, 25(3):579–588, 2006.

[29] C. Loop, C. Zhang, and Z. Zhang. Real-time High-resolution Sparse Voxelization with Application to Image-based Modeling. In *Proc. of the High-Performance Graphics Conference*, pp. 73–79, 2013.

[30] W. E. Lorensen and H. E. Cline. Marching Cubes: A High Resolution 3D Surface Construction Algorithm. In *Proc. of the 14th Annual Conf. on Computer Graphics and Interactive Techniques*, pp. 163–169, 1987.

[31] A. Maimone, J. Bidwell, K. Peng, and H. Fuchs. Enhanced personal autostereoscopic telepresence system using commodity depth cameras. *Computers & Graphics*, 36(7):791 – 807, 2012.

[32] A. Maimone and H. Fuchs. Real-time volumetric 3D capture of room-sized scenes for telepresence. In *Proc. of the 3DTV-Conference*, 2012.

[33] S. Mangiante, G. Klas, A. Navon, Z. GuanHua, J. Ran, and M. D. Silva. VR is on the Edge: How to Deliver 360° Videos in Mobile Networks. In *Proc. of the Workshop on Virtual Reality and Augmented Reality Network*, pp. 30–35, 2017.

[34] D. Molyneaux, S. Izadi, D. Kim, O. Hilliges, S. Hodges, X. Cao, A. Butler, and H. Gellersen. Interactive Environment-Aware Handheld Projectors for Pervasive Computing Spaces. In *Proc. of the Int. Conf. on Pervasive Computing*, pp. 197–215, 2012.

[35] A. Mossel and M. Kröter. Streaming and exploration of dynamically changing dense 3d reconstructions in immersive virtual reality. In *Proc. of IEEE Int. Symp. on Mixed and Augmented Reality*, pp. 43–48, 2016.

[36] J. Mulligan and K. Daniilidis. View-independent scene acquisition for tele-presence. In *Proc. IEEE and ACM Int. Symp. on Augmented Reality*, pp. 105–108, 2000.

[37] R. A. Newcombe et al. KinectFusion: Real-Time Dense Surface Mapping and Tracking. In *Proc. of IEEE Int. Symp. on Mixed and Augmented Reality*. IEEE, 2011.

[38] R. A. Newcombe, D. Fox, and S. M. Seitz. DynamicFusion: Reconstruction and tracking of non-rigid scenes in real-time. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 343–352, 2015.

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

[40] S. Orts-Escolano et al. Holoportation: Virtual 3D Teleportation in Real-time. In *Proc. of the Annual Symp. on User Interface Software and Technology*, pp. 741–754, 2016.

[41] B. Petit, J.-D. Lesage, C. Menier, J. Allard, J.-S. Franco, B. Raffin, E. Boyer, and F. Faure. Multicamera Real-Time 3D Modeling for Telepresence and Remote Collaboration. *Int. Journal of Digital Multimedia Broadcasting*, 2010.

[42] PresenterMedia. PowerPoint Templates, 3D Animations, and Clipart. https://presentermedia.com/, 2009. Accessed: 2019-01-29.

[43] H. Roth and M. Vona. Moving volume kinectfusion. In *Proc. of the British Machine Vision Conference*, pp. 112.1–112.11, 2012.

[44] T. Schöps, J. Engel, and D. Cremers. Semi-dense visual odometry for ar on a smartphone. In *Int. Symp. on Mixed and Augmented Reality*, 2014.

[45] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A Benchmark for the Evaluation of RGB-D SLAM Systems. In *Proc. of the Int. Conf. on Intelligent Robot Systems*, 2012.

[46] T. Tanikawa, Y. Suzuki, K. Hirota, and M. Hirose. Real world video avatar: Real-time and real-size transmission and presentation of human figure. In *Proc. of the Int. Conf. on Augmented Tele-existence*, pp. 112–118, 2005.

[47] H. Towles, W. Chen, R. Yang, S. Kum, H. Fuchs, N. Kelshikar, J. Mulligan, K. Daniilidis, C. C. Hill, L. Holden, B. Zeleznik, A. Sadagic, and J. Lanier. 3D Tele-Collaboration Over Internet2. In *Proc. of the Int. Workshop on Immersive Telepresence*, 2002.

[48] T. T. Tran, M. Giraud, and J.-S. Varré. Perfect Hashing Structures for Parallel Similarity Searches. *IEEE Int. Parallel and Distributed Processing Symposium Workshop*, pp. 332–341, 2015.

[49] R. Vasudevan, G. Kurillo, E. Lobaton, T. Bernardin, O. Kreylos, R. Bajcsy, and K. Nahrstedt. High-Quality Visualization for Geographically Distributed 3-D Teleimmersive Applications. *IEEE Trans. on Multimedia*, 13(3):573–584, 2011.

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

[51] 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 Int. Journal of Robotics Research*, 34(4-5):598–626, 2015.

[52] B. G. Witmer and M. J. Singer. Measuring Presence in Virtual Environments: A Presence Questionnaire. *Presence: Teleoper. Virtual Environ.*, 7(3):225–240, 1998.

[53] G. Zumbach and U. Müller. Operators on inhomogeneous time series. *Int. Journal of Theoretical and Applied Finance*, 4(01):147–177, 2001.

Publication:
“Efficient 3D Reconstruction and Streaming for Group-Scale Multi-Client Live Telepresence”

Patrick Stotko, Stefan Krumpen, Michael Weinmann, and Reinhard Klein

IEEE International Symposium on Mixed and Augmented Reality (ISMAR)

2019

DOI: 10.1109/ISMAR.2019.00018

Efficient 3D Reconstruction and Streaming for Group-Scale Multi-Client Live Telepresence

Patrick Stotko* Stefan Krumpen† Michael Weinmann‡ Reinhard Klein§

University of Bonn

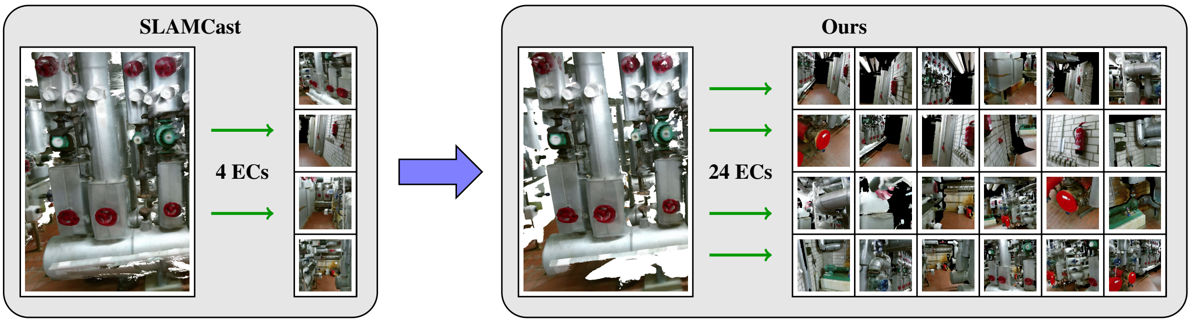


Figure 1: Illustration of our novel highly scalable multi-client live telepresence system. While previous approaches are limited to a low number of up to 4 remote exploration clients, our system is capable of providing an immersive telepresence experience within a live-captured high-quality scene reconstruction to more than 24 clients simultaneously without introducing further latency.

Abstract

Sharing live telepresence experiences for teleconferencing or remote collaboration receives increasing interest with the recent progress in capturing and AR/VR technology. Whereas impressive telepresence systems have been proposed on top of on-the-fly scene capture, data transmission and visualization, these systems are restricted to the immersion of single or up to a low number of users into the respective scenarios. In this paper, we direct our attention on immersing significantly larger groups of people into live-captured scenes as required in education, entertainment or collaboration scenarios. For this purpose, rather than abandoning previous approaches, we present a range of optimizations of the involved reconstruction and streaming components that allow the immersion of a group of more than 24 users within the same scene – which is about a factor of 6 higher than in previous work – without introducing further latency or changing the involved consumer hardware setup. We demonstrate that our optimized system is capable of generating high-quality scene reconstructions as well as providing an immersive viewing experience to a large group of people within these live-captured scenes.

Index Terms: Human-centered computing—Human computer interaction (HCI)—Interaction paradigms—Virtual reality; Human-centered computing—Human computer interaction (HCI)—Interaction paradigms—Collaborative interaction; Computing methodologies—Computer graphics—Graphics systems and interfaces—Virtual reality; Computing methodologies—Computer vision—Computer vision problems—Reconstruction

1 Introduction

The rapidly increasing potential of AR/VR technology has led to several highly advanced telepresence applications such as teleconferencing in room-scale environments [4, 24] or the exploration of places – that may vary from the users' local physical environment – for live-captured scenes beyond room-scale [27] and for remote collaboration purposes. To meet the critical success factors of an immersive telepresence experience for on-the-fly captured 3D data as required by these scenarios, these systems impose strong demands regarding the reconstruction and streaming speed as well as the visual quality of the acquired scene. Furthermore, the interactive exploration within the scene requires rendering at high framerates and low latency to avoid motion sickness. This means that all of the involved processing steps including 3D scene capture, data transmission and visualization have to be achieved in real-time, while taking the typically available network bandwidth and client-side compute hardware into account. Previous teleconferencing systems [4, 24, 28] were designed to capture a fixed region of interest based on expensive well-calibrated acquisition setups involving statically mounted cameras. In contrast, the live telepresence system by Stotko et al. [27] is tailored to the acquisition of scenes beyond such a fixed size defined by the setup and involves portable, consumer-grade capture hardware. As a result, efficiently representing and transmitting the scene to visualization devices is significantly harder. While further work has been spent on parallelized capturing [6], the goal of immersing a large number of people into the same live-captured environment while allowing them to interact with each other for e.g. remote collaboration and exploration scenarios, to the best of our knowledge, has not received a lot of attention so far. In particular, the major challenge is given by the accurate reconstruction and transmission of 3D models while keeping the computational burden as well as the memory and streaming requirements as low as possible, thus, minimizing the amount of unnecessary or unreliable model data resulting from noise and outliers in the captured input data.

In this paper, we address the scalability of live telepresence systems to the immersion of whole groups of (more than 24) people without introducing further latency as required for education, entertainment and collaboration scenarios (see Fig. 1). For this purpose, rather than developing new techniques for 3D capture and data transmission, we demonstrate how existing well-established systems for 3D reconstruction and streaming can be optimized to significantly increase the scalability of live telepresence systems under strong

*e-mail: stotko@cs.uni-bonn.de
†e-mail: krumpen@cs.uni-bonn.de
‡e-mail: mw@cs.uni-bonn.de
§e-mail: rk@cs.uni-bonn.de

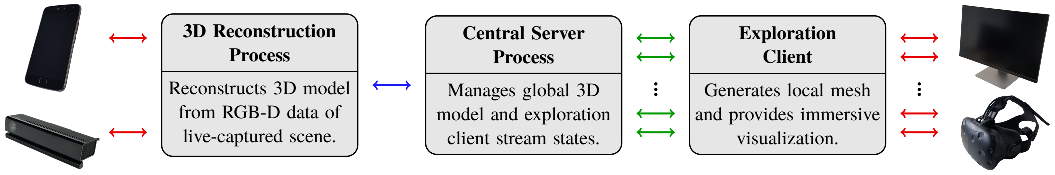

3D Reconstruction Process
Reconstructs 3D model from RGB-D data of live-captured scene.
Central Server Process
Manages global 3D model and exploration client stream states.
Exploration Client
Generates local mesh and provides immersive visualization.

Figure 2: Overview of the major components of state-of-the-art live multi-client live telepresence systems. RGB-D image data acquired by a single camera device are streamed to a cloud server (red arrows) where a global 3D scene model is reconstructed in a dedicated 3D reconstruction process. This scene model is then passed to the central server process (blue arrows) which also runs on the cloud server and manages a bandwidth-optimized version of the model as well as the client states. Large groups of people (more than 24 in our system), each running an exploration client on their local hardware, can independently request parts of the reconstructed global scene model (green arrows) and render the locally generated mesh on their display devices (red arrows).

constraints regarding latency and bandwidth. We demonstrate that our extensions result in compact high-quality 3D reconstructions and finally allow the immersion of more than 24 people within the same live-captured scene (beyond room-scale), thereby significantly exceeding the number of immersed persons in previous approaches [27] without adding or exchanging hardware components.

In summary, the key contributions of this work are (1) an efficient novel set of filters designed to optimize the performance and scalability of current state-of-the-art telepresence systems at the example of the SLAMCast system [27], (2) an adaption of the telepresence-specific filters to standalone volumetric 3D reconstruction and (3) a comprehensive evaluation of the beneficial effect of the proposed set of filters regarding scalability, latency and visual quality.

2 Related Work

Telepresence applications for sharing live experiences rely on real-time 3D scene capture. For this purpose, the underlying scene representation, where the scene is reconstructed based on the fusion of the incoming sensor data, is of particular importance. Well-established representations include surface modeling in the form of implicit truncated signed distance fields (TSDFs). Early real-time volumetric reconstruction approaches [9, 21] are based on storing the scene model in a uniform grid. This results in high memory requirements as the data structure is not adapted according to the local presence of a surface. To improve the scalability to large-scale scenes, further work exploited the sparsity in the TSDF representation, e.g. based on moving volume techniques [26, 31], representing scenes in terms of blocks of volumes that follow dominant planes [8] or storing TSDF values only near the actual surface areas [1, 12, 23]. The individual blocks can be managed using tree structures or hash maps as proposed by Nießner et al. [23] and respective optimizations [12, 13, 25]. Furthermore, the replacement of the TSDF representation by a high-resolution binary voxel grid has also been considered by Reichl et al. [25] to improve the scalability and reduce the memory requirements. Recent extensions include the detection of loop closures [2, 11, 16] to reduce drift artifacts in camera localization as well as multi-client collaborative acquisition and reconstruction of static scenes [6].

This progress in real-time capturing enabled the development of various telepresence applications. Early telepresence systems [5, 9, 10, 17–19] were designed for room-scale environments and faced the problems of a limited reconstruction quality due to high sensor noise and a reduced resolution. Relying on an expensive capturing setup with several cameras, GPUs and desktop computers, the Holoportation system [24] was designed for high-quality real-time reconstruction of a dynamic room-scale environment based on the Fusion4D system [3] as well as real-time data transmission. This has been complemented with AR/VR systems to allow immersive end-to-end teleconferencing. In contrast, interactive telepresence for individual remote users within live-captured static scenes has been addressed by Mossel and Kröter [20] based on voxel block hashing [12, 23]. The limitations of this system regarding high bandwidth requirements, the immersion of only a single remote user into the captured scenarios as well as network interruptions leading to loss of scene parts that are reconstructed in the meantime have been overcome in the recent SLAMCast system [27]. However, the scalability to immersing large groups of people into on-the-fly captured scenes has not been achieved so far. In this paper, we directly address this problem by several modifications to the major components involved in telepresence systems.

3 System Outline

Akin to previous work, we build our scalable multi-client telepresence system on top of a volumetric scene representation in terms of voxel blocks, i.e. blocks of $8^3 = 512$ voxels. This approach has been well-established by previous investigations in the context of real-time reconstruction [1, 2, 9, 11–13, 21–23, 30, 31] and telepresence [20, 24, 27]. As shown in Fig. 2, current state-of-the-art telepresence systems involving live-captured scenarios rely on the core components of (1) a real-time 3D reconstruction process, (2) a central server process as well as (3) exploration clients. RGB-D images captured by a single camera are streamed to the reconstruction process that runs on a cloud server and allows on-the-fly camera localization and scene capture via volumetric fusion. The reconstructed scene data is then passed to the central server process that manages a bandwidth-optimized version of the global model as well as the streaming of these data according to requests by connected exploration clients. Each exploration client integrates the transmitted scene parts into a locally generated mesh that can be interactively explored with VR devices on their local computers. In the following, we focus on the extension of such live telepresence systems to the immersion of larger groups of remote users into a live-captured scene at the example of the SLAMCast system [27]. This requires the optimization of the reconstruction (see Sect. 4) and the central server processes (see Sect. 5). In contrast, the exploration client receives the compressed and optimized scene representation and is already capable of providing an immersive viewing experience at the remote user's site.

4 Optimization of the 3D Reconstruction Process

Since our optimizations are not particularly restricted to the reconstruction process used in the SLAMCast system, we show their application to volumetric 3D reconstruction approaches in general and provide an overview of the respective pipeline (see Fig. 3). Here, the surface is represented in terms of implicit truncated signed distance fields (TSDFs) and stored as a sparse unordered set of voxel blocks using spatial hashing [2, 11, 12, 23, 25]. Input to the reconstruction pipeline is an incremental stream of RGB-D images which is processed in an online fashion. First, the current RGB-D frame is preprocessed where camera-specific distortion effects are removed and a normal map is computed from the depth data. Afterwards, the current camera pose is estimated either using frame-to-model

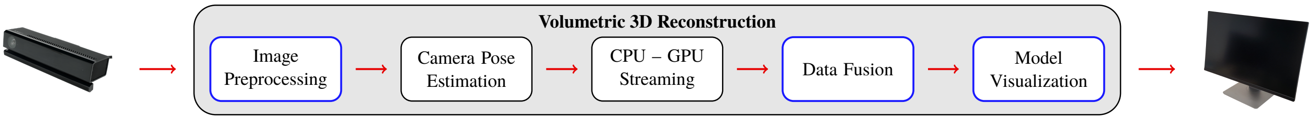


Figure 3: General volumetric 3D reconstruction pipeline. Our set of efficient filters designed to improve the performance and scalability of the state-of-the-art live telepresence systems can also be applied to the components of standalone 3D reconstruction (highlighted).

tracking [9, 12, 21, 23, 25, 29, 31] (as also used in the SLAMCast system) or using bundle adjustment for globally-consistent reconstruction [2, 11]. Using this pose, non-visible voxel block data are streamed out to CPU memory whereas visible blocks in CPU memory are streamed back into GPU memory [12, 16, 23, 25]. In the next step, new voxel blocks are allocated in the volume and the RGB-D data are fused into the volumetric model. Finally, a novel view depicting the current state of the reconstruction is generated using raycasting to provide a live feedback to the user during capturing.

4.1 Image Preprocessing

We improve the robustness of the acquired RGB-D data by filtering potentially unreliable data from the depth map. A further benefit of this operation is the resulting more compact scene model representation. Inspired by previous work [31], we discard samples d located on stark depth discontinuities by considering the deviations to the depth values d_i in a 7×7 neighborhood $\boldsymbol{N}(d)$. Due to the limited resolution and the overall noise characteristics of the sensor, such samples are likely to be outliers and might largely deviate from the true depth values. We extend this filter by further discarding samples d with a significant amount of missing data in their local neighborhood. In such regions, which may not only contain depth discontinuities, the depth measurements are also susceptible of being unreliable. Thus, we consider the set

$$\boldsymbol{D}_o = \{d \mid \exists i \in \boldsymbol{N}(d)\colon |d - d_i| > c_d \vee |\boldsymbol{N}_o(d)| > c_h \cdot |\boldsymbol{N}(d)|\} \quad (1)$$

as outliers where c_d and c_h are user-defined thresholds, $\boldsymbol{N}(d)$ denotes the neighborhood of the depth sample d and $\boldsymbol{N}_o(d)$ the set of neighboring pixels with no valid depth data. These outliers affect the overall reconstruction quality as well as the model compactness.

4.2 Data Fusion

Although potentially unreliable data around stark depth discontinuities have been filtered out during the preprocessing step, there are still samples, e.g. around small discontinuities, that do not contribute to the reconstruction and negatively affect the model compactness and streaming performance. In the voxel block allocation step, these unreliable data unnecessarily enlarge the global truncation region around the unknown surface since all voxel blocks located within the local truncation region around the respective depth samples are considered during allocation. Traditional approaches tried to remove these blocks afterwards using a garbage collection [23] which requires a costly analysis of the voxel data. In contrast, we propose a novel implicit filter which reduces the amount of unnecessary block allocations. By considering only every c_a-th pixel per column and row, where c_a is a user-defined control parameter, the depth image is virtually downsampled and the likelihood for an over-sized global truncation region is significantly reduced. Furthermore, this reduces the number of processed voxels during data fusion which greatly speed-ups the reconstruction and reduces the amount of blocks that are later queued for streaming to the server. Note that this downsampling is only performed during allocation whereas the whole depth image is still used for data fusion to employ TSDF-based regularization. In the context of globally-consistent 3D reconstruction using bundle-adjusted submaps [6, 11], our filter improves the compactness of the respective submap into which the RGB-D data are fused whereas the fusion of the more compact submaps into a single global model would be performed as in previous work.

Algorithm 1 Our optimized server voxel block data integration

```
Input: Received TSDF voxel block positions $\boldsymbol{P}^{TSDF}$ and voxel data $\boldsymbol{V}^{TSDF}$
Output: Voxel block position list $\boldsymbol{P}^{MC}$ for updating the stream sets
  $\boldsymbol{M}^{TSDF} \leftarrow$ allocateBlocks($\boldsymbol{P}^{TSDF}$)
  $\boldsymbol{M}^{TSDF} \leftarrow$ copyVoxelData($\boldsymbol{V}^{TSDF}$)
  $\boldsymbol{P}^{MC} \leftarrow$ createBlockUpdateSet($\boldsymbol{P}^{TSDF}$)
  $\boldsymbol{V}^{MC}, \boldsymbol{F}^{MC} \leftarrow$ computeVoxelData($\boldsymbol{P}^{MC}, \boldsymbol{M}^{TSDF}$)
  $\boldsymbol{M}^{MC} \leftarrow$ allocateNonEmptyBlocks($\boldsymbol{P}^{MC}, \boldsymbol{F}^{MC}$)
  $\boldsymbol{M}^{MC} \leftarrow$ copyNonEmptyVoxelData($\boldsymbol{V}^{MC}, \boldsymbol{F}^{MC}$)
  $\boldsymbol{M}^{MC} \leftarrow$ pruneEmptyBlocks($\boldsymbol{P}^{MC}, \boldsymbol{F}^{MC}$)
  $\boldsymbol{P}^{MC} \leftarrow$ pruneBlockUpdateSet($\boldsymbol{P}^{MC}, \boldsymbol{P}_A^{MC}, \boldsymbol{F}^{MC}$)
  $\boldsymbol{P}_A^{MC} \leftarrow$ updateNonEmptyBlockSet($\boldsymbol{P}_A^{MC}, \boldsymbol{P}^{MC}$)
```

4.3 Model Visualization

In order to provide a decent live preview of the current model state, the generation of such model views should preserve all the relevant scene information while suppressing noise as much as possible. Furthermore, if frame-to-model tracking is used to estimate the current camera pose, this is also crucial to allow a robust alignment. We propose a Marching Cubes (MC) voxel block pruning approach which will be described in more detail in Sect. 5, as it has been carefully designed for the central server process. Here, we show an adaption of this contribution to standalone volumetric 3D reconstruction where the model is stored implicitly using TSDF voxels. Each TSDF voxel stores a TSDF value $D \in [-1;1]$ and fusion weight $W \in [0;255]$ (both compressed using 16-bit linear encoding [12]) as well as a 24-bit color $C \in [0;255]^3$. Inspired by the garbage collection approach of point-based reconstruction techniques [14], we ignore TSDF voxels for raycasting and triangle generation which are currently considered unstable. These voxels contain only very few, possibly unreliable observations from the input data, so their fusion weight falls below a user-defined threshold c_w:

$$\boldsymbol{V}_o^{TSDF} = \{(D, W, C) \mid W < c_w\} \quad (2)$$

However, in contrast to previous garbage collection approaches [14, 23], we do not remove these voxel blocks but only ignore them. This avoids accidental removal of blocks that might become stable at a future time when this scene part is also partially stored in a different submap or revisited by the user or another client in multi-client acquisition setups [6, 11]. Furthermore, by ignoring unstable data, the raycasted view will also be consistent with the exploration client's version of the 3D model.

5 Optimization of the Central Server Process

Beyond optimizations in the reconstruction process, the scalability of a live telepresence system also relies on the optimization of its central server process that takes care of managing the reconstructed global scene model as well as the stream states and requests by connected exploration clients. In this regard, we show respective optimizations at the example of the recently published SLAMCast system [27]. In comparison to the standard voxel block data integration at the server side, we propose a further filtering step which discards empty or unstable voxel blocks that contain only very few or none observations from the input RGB-D image data. This significantly improves the streaming performance and scalability and allows the immersion of groups of people. The individual steps of our optimized integration approach are shown in Algorithm 1.

Similar to the original SLAMCast system, we first integrate the TSDF voxel block positions $\boldsymbol{P}^{TSDF}$ and voxel data $\boldsymbol{V}^{TSDF}$ into the global TSDF voxel block model $\boldsymbol{M}^{TSDF}$ of the central server process. Afterwards, we update the global MC voxel block model $\boldsymbol{M}^{MC}$ which is optimized for streaming and stores a Marching Cubes index $I \in [0;255]$ as well as a 24-bit color $C \in [0;255]^3$ in each MC voxel. For this purpose, we create the set $\boldsymbol{P}^{MC}$ of MC voxel block positions requiring an update as well as a set of flags $\boldsymbol{F}^{MC}$ and the respective MC voxel data $\boldsymbol{V}^{MC}$ by performing the Marching Cubes algorithm on the corresponding TSDF voxels [15]. The flags $\boldsymbol{F}^{MC}$ indicate whether a block will generate reliable triangles and are constructed by analyzing the Marching Cubes indices I of the MC voxels as well as the fusion weight W of the corresponding TSDF voxels. Therefore, the following set $\boldsymbol{V}_o^{MC}$ of voxels either does not contain surface information in terms of triangles or would generate unstable triangle data:

$$\boldsymbol{V}_o^{MC} = \{(I, C) \mid I = 0 \vee I = 255 \vee W < c_w\} \qquad (3)$$

We only allocate those blocks in the MC voxel block model $\boldsymbol{M}^{MC}$ that are flagged and prune blocks that are currently not flagged. This minimizes the amount of scene data that are streamed to the exploration clients. Finally, we integrate the generated MC voxel data $\boldsymbol{V}^{MC}$ of the flagged blocks. We do not prune the TSDF voxel block model $\boldsymbol{M}^{TSDF}$ which would otherwise lead to potential artifacts, i.e. missing geometry at block boundaries, since currently empty blocks might be needed for future updates.

In contrast to the MC voxel block model, pruning the list of updated MC voxel block positions $\boldsymbol{P}^{MC}$ in the same way would introduce artifacts at the exploration client side since they may already have received a previous version of blocks that have been pruned in the meantime. To properly handle updates, we manage the update set $\boldsymbol{P}_A^{MC}$ containing all voxel block positions that were considered for streaming in the past. We generate the list of updated MC voxel blocks by only considering the ones which either generated triangles in the past or with the current update. Finally, after the MC voxel blocks have been integrated into the volume and the list of updated block positions has been generated, we update the set $\boldsymbol{P}_A^{MC}$ by inserting all currently integrated voxel block positions.

6 Evaluation

We tested our highly scalable telepresence system on a variety of different datasets and analyzed several aspects such as system scalability, streaming latency and visual quality. For a quantitative comparison of the proposed contributions, we considered the following variants of our system:

- **Base (B)**: Our 3D reconstruction and streaming system with deactivated filtering contributions, yielding equivalent performance to SLAMCast [27].
- **Base + Depth Discontinuity Filter (B+DDF)**: The base approach with an additional depth map filtering at discontinuities with c_h = 0.25, c_d = 0.2m (see Sect. 4.1).
- **Base + Voxel Block Allocation Downsampling (B+VBAD)**: The base approach with an additional virtual downsampling at the voxel block allocation stage with c_a = 4 (see Sect. 4.2).
- **Base + MC Voxel Block Pruning (B+MCVBP)**: The base approach with an additional pruning of empty MC voxel blocks at the server side with c_w = 2.0 (see Sect. 4.3 and Sect. 5).
- **Ours**: Our approach incorporating all filtering contributions.

Table 1: Maximum number of exploration clients (ECs) that the server can handle without any delay compared to a single client. Instead of using different package sizes with a fixed request rate of 100Hz, we use a fixed size of 512 and vary the rate accordingly to demonstrate the highest possible scalability. If empty MC voxel block pruning is used (B+MCVBP and Ours), the sizes of the TSDF and MC voxel block models differ and we list both ($\boldsymbol{M}^{MC}/\boldsymbol{P}_A^{MC}(\boldsymbol{M}^{TSDF})$).

Approach	Dataset	Max. ECs	Request Rate [Hz]	Model Size [# $\times 10^3$ MC Voxel Blocks]
B	*lounge*	5	100	314
	copyroom	9	50	228
	heating_room	3	100	850
	pool	5	100	590
	lr kt2	1	200	834
B+DDF	*lounge*	9	50	270
	copyroom	10	50	230
	heating_room	9	50	443
	pool	10	50	379
	lr kt2	10	50	227
B+VBAD	*lounge*	8	50	264
	copyroom	5	100	226
	heating_room	4	100	622
	pool	8	50	446
	lr kt2	1	200	550
B+MCVBP	*lounge*	21	12	47 / 51 (314)
	copyroom	9	50	65 / 75 (298)
	heating_room	8	25	120 / 127 (850)
	pool	13	25	104 / 108 (590)
	lr kt2	7	25	64 / 64 (834)
Ours	*lounge*	25	12	44 / 47 (240)
	copyroom	18	25	57 / 67 (202)
	heating_room	27	12	90 / 94 (352)
	pool	28	12	95 / 99 (317)
	lr kt2	26	12	53 / 55 (201)

The filter sizes and thresholds as described above were determined empirically using several datasets. For validation, we used different real-world datasets recorded with an ASUS Xtion Pro (*lounge*, *copyroom*) [32] and a Kinect v2 (*heating_room*, *pool*) [27] as well as synthetic data (*lr kt2* with simulated noise) [7]. Throughout the experiments, we used three computers where each of them takes the role of one part of the telepresence system, i.e. 3D reconstruction process (RC), central server process (S) and exploration client (EC). All computers were equipped with an Intel Core i7-4930K CPU and 32GB RAM and a NVIDIA GTX 1080 GPU with 8GB VRAM and connected via a local network. We replaced the exploration client by a benchmark client which starts requesting voxel blocks with a fixed frame rate of 100Hz when the reconstruction process starts. Furthermore, the reconstruction process uses a fixed reconstruction speed of 30Hz matching the framerate of the used datasets. We set the voxel size to 5mm as well as the truncation region to 60mm and used hash map/set sizes of 2^{20} and 2^{22} buckets as well as GPU and CPU voxel block pool sizes of 2^{19} and 2^{20} blocks, thereby following previous work [27].

6.1 System Scalability

In this section, we will evaluate the scalability of our system in comparison to the baseline SLAMCast approach (see Table 1). In contrast to the following evaluations, the benchmark client discards the received data which allows for running all benchmark clients on a single computer without an overhead. Furthermore, rather than lowering the package size, we used a fixed package size of 512 voxel blocks and lower the request rate accordingly. This significantly reduced the constant overheads of kernel calls and memory copies and introduces only a minimal delay in the range of milliseconds which made it the preferred setting for handling a large number of clients. For an appropriate choice of the streaming rate, we deter-

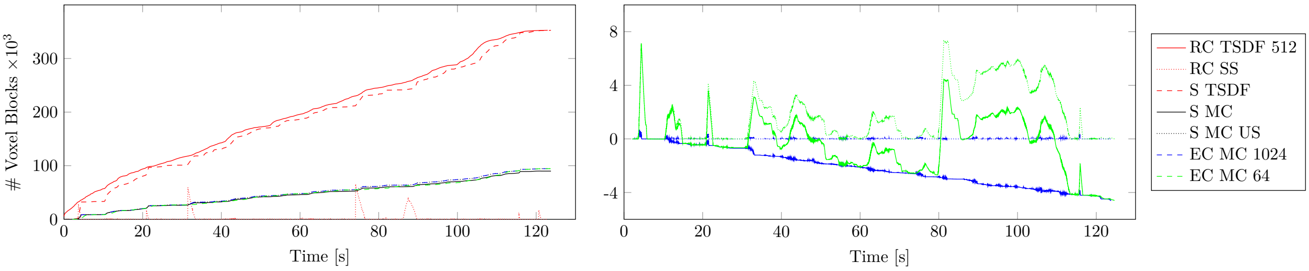


Figure 4: Streaming progress and latency between server (S) and exploration client (EC) over time for the *heating_room* dataset using our full system. Left: Absolute model sizes for the highest and lowest chosen package size. Right: Relative size differences between S and EC (w.r.t. model size S MC and update set size S MC US).

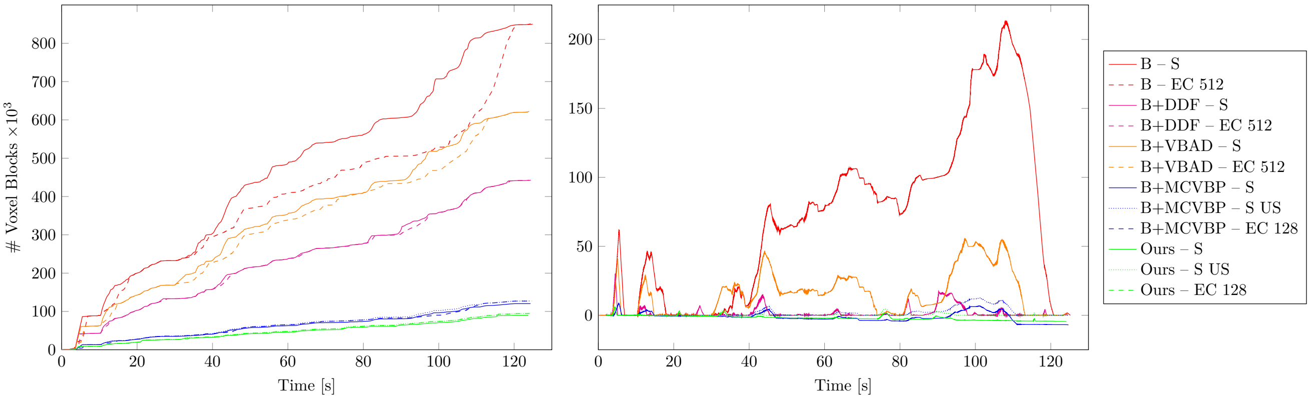


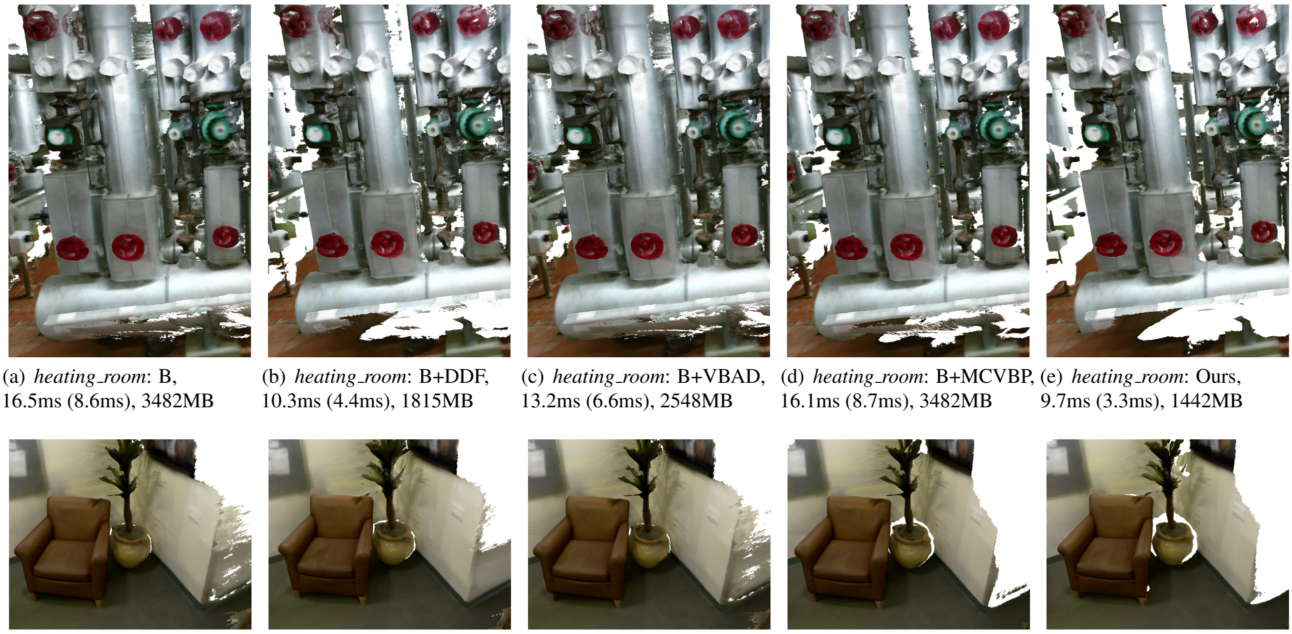
Figure 5: Streaming progress and latency between server (S) and exploration client (EC) over time for the *heating_room* dataset for each system variant. Left: Absolute model sizes. Right: Relative size differences between S and EC.

mined the lowest package size which still allows the benchmark client to retrieve the whole model with an acceptable delay of at most one second (see supplemental material for a detailed analysis). Then, we measured the maximum number of benchmark clients that the server could handle without introducing a further delay. While the original SLAMCast system was only able to handle around 3-5 clients in general, both filters at the reconstruction side (B+DDF and B+VBAD) raised this limit to up to 10 clients. Since there is a tracking loss at the end of the *copyroom* sequence resulting in a slightly higher delay, a higher request rate was chosen and the scalability decreased accordingly. Although the number of MC voxel blocks is significantly lower after pruning (B+MCVBP), we observed that the general performance is similar to the depth discontinuity filter approach. Here, the TSDF voxel block model has the same size as in the base approach and is, hence, considerably larger than in the other approaches. In contrast, our full system reduces the request rate requirements to 12Hz for most scenes making it the preferred choice for this parameter. This significantly improves its scalability to more than 24 clients in all scenes which is sufficient for applications in education, entertainment or collaboration scenarios.

6.2 Latency and Streaming Progress Analysis

In addition to the scalability analysis, we also measured the streaming latency over time (see Fig. 4). Similar to the original SLAMCast approach, our system has a small delay between the reconstruction process and the server process due to the shared streaming strategy. However, our optimized server model prunes unreliable or irrelevant blocks which results in a very low latency between the server and the exploration client. We also compared the latency between the largest and smallest chosen package size, i.e. 1024 and 64 blocks/request. Here, the model size of the exploration client is close to the size of the server's update set $\boldsymbol{P}_A^{MC}$ indicating a very fast and low-latent streaming while the gap to the minimal size of the server model $\boldsymbol{M}^{MC}$ increases over time. Note that these two sizes are the bounds for the exploration client's model size and clients which have reconnected, e.g. due to network outages, will receive a slightly more compact model closer to the lower bound. Reducing the package size from 1024 to 64 blocks significantly reduces the bandwidth requirements (see supplemental material for a detailed analysis) and leads to a slightly worse latency when the reconstruction process queues the currently visible voxel blocks for streaming.

In Fig. 5, we also compared the different system variants regarding streaming progress and latency. For a fair comparison between the approaches, the package size is chosen such that the mean bandwidths are similar, i.e. around 15Mbit/s. Here, we also considered the size of the update set $\boldsymbol{P}_A^{MC}$ in addition to size of the server model $\boldsymbol{M}^{MC}$ when empty MC voxel block pruning is enabled (B+MCVBP and Ours). In these scenarios, the number of voxel blocks transmitted to the exploration client bound by these two sizes is typically close to the upper bound $\boldsymbol{P}_A^{MC}$. In comparison to the baseline, both filtering approaches at the reconstruction side (B+DDF and B+VBAD) reduce the latency significantly. Similar results can be seen when empty MC voxel blocks are pruned (B+MCVBP). Whereas all of these approaches still introduce a noticeable delay at the time steps 40s and 90-100s, our full system is capable of streaming the reconstructed model with almost no delay across the whole sequence. Additional results regarding bandwidth and streaming latency over time are provided in the supplemental material.



(a) *heating_room*: B, 16.5ms (8.6ms), 3482MB (b) *heating_room*: B+DDF, 10.3ms (4.4ms), 1815MB (c) *heating_room*: B+VBAD, 13.2ms (6.6ms), 2548MB (d) *heating_room*: B+MCVBP, 16.1ms (8.7ms), 3482MB (e) *heating_room*: Ours, 9.7ms (3.3ms), 1442MB

(f) *lounge*: B, 10.9ms (5.0ms), 1286MB (g) *lounge*: B+DDF, 10.0ms (4.4ms), 1106MB (h) *lounge*: B+VBAD, 10.0ms (5.4ms), 1081MB (i) *lounge*: B+MCVBP, 10.9ms (5.2ms), 1286MB (j) *lounge*: Ours, 9.7ms (4.3ms), 983MB

Figure 6: Comparison of visual quality, mean runtime (and standard deviation) as well as memory requirements for each system variant. All individual contributions reduced the amount of reconstruction artifacts while improving the overall reconstruction performance.

6.3 Visual Quality

In order to demonstrate the benefit for standalone volumetric 3D reconstruction, we also provide a qualitative comparison regarding the visual quality of the reconstructed 3D models as well as the respective runtime and memory requirements for the individual system variants (see Fig. 6). In general, all approaches generated detailed and accurate 3D models from the noisy RGB-D input data. However, without filtering, there might be some artifacts around depth discontinuities as well as in regions which have not been fully observed by the camera. These artifacts affect the overall visual experience and lead to high runtime and memory requirements. Using virtual downsampling at the voxel block allocation stage (B+VBAD), we obtain almost identical 3D models but the computational burden is significantly lower since the number of empty blocks within the model is reduced. In contrast, filtering depth samples at depth discontinuities (B+DDF) or unreliable triangle data during Marching Cubes (B+MCVBP) reduces the amount of artifacts in the aforementioned regions. Note that in standalone 3D reconstruction, voxel block pruning (B+MCVBP) mainly affects the triangulation step at the end of the capturing session which leads to results similar to the base approach regarding runtime and memory. Our full system enhances the visual quality even further and almost completely removes artifacts without sacrificing the overall model completeness. Here, we observe improvements of 10-40% and 25-60% for the runtime and memory footprint respectively depending on the scene. The objects in the *lounge* scene have been captured at a much smaller distance and from more angles than in the *heating_room* scene which leads to less unreliable input data and, hence, a lower impact of our outlier filtering approach. Additional performance measurements and results are provided in the supplemental material. In the context of live remote collaboration, a slightly less complete model can be beneficial and helps to identify regions that still need to be captured and reliably reconstructed. This, in turn, might even increase the model completeness and accuracy since the scene is more thoroughly acquired by the user.

6.4 Limitations

Despite the significant improvements in terms of scalability, latency and visual quality, our system still has some limitations. Since our work is based on the SLAMCast system, misalignments within the reconstruction might occur due to fast camera movement. While this problem has been addressed by loop-closure techniques [2, 11], their integration into live telepresence systems is still highly challenging. Furthermore, too aggressive virtual downsampling during voxel block allocation might lead to holes in the final model when some blocks covering distant objects are always skipped and, hence, never allocated. However, this is only problematic for long-range devices whereas typical RGB-D cameras have a smaller range of up to 5 meter which is still sufficient for most scenarios.

7 Conclusion

We presented a highly scalable multi-client live telepresence system which allows immersing a large number of people into a live-captured environment. For this purpose, we used well-established systems and proposed several optimizations regarding scalability, latency, and visual quality. While our contributions are designed with the telepresence system in mind, we also show their application to standalone volumetric 3D reconstruction approaches. As demonstrated in a comprehensive evaluation, our novel system allows the immersion of more than 24 people within the same scene using consumer hardware.

Acknowledgments

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

[2] A. Dai, M. Nießner, M. Zollhöfer, S. Izadi, and C. Theobalt. Bundle-Fusion: Real-time Globally Consistent 3D Reconstruction using On-the-fly Surface Reintegration. *ACM Trans. Graph.*, 36(3):24, 2017.

[3] M. Dou et al. Fusion4D: Real-time Performance Capture of Challenging Scenes. *ACM Trans. Graph.*, 35(4):114:1–114:13, 2016.

[4] A. J. Fairchild, S. P. Campion, A. S. García, R. Wolff, T. Fernando, and D. J. Roberts. A Mixed Reality Telepresence System for Collaborative Space Operation. *IEEE Trans. on Circuits and Systems for Video Technology*, 27(4):814–827, 2016.

[5] H. Fuchs, A. State, and J. Bazin. Immersive 3D Telepresence. *Computer*, 47(7):46–52, 2014.

[6] S. Golodetz, T. Cavallari, N. A. Lord, V. A. Prisacariu, D. W. Murray, and P. H. S. Torr. Collaborative Large-Scale Dense 3D Reconstruction with Online Inter-Agent Pose Optimisation. *IEEE Trans. on Visualization and Computer Graphics*, 24(11):2895–2905, Nov 2018.

[7] A. Handa, T. Whelan, J. McDonald, and A. J. Davison. A Benchmark for RGB-D Visual Odometry, 3D Reconstruction and SLAM. In *Proc. of the Int. Conf. on Robotics and Automation*, pp. 1524–1531, 2014.

[8] P. Henry, D. Fox, A. Bhowmik, and R. Mongia. Patch Volumes: Segmentation-Based Consistent Mapping with RGB-D Cameras. In *Int. Conf. on 3D Vision*, 2013.

[9] S. Izadi et al. KinectFusion: Real-time 3D Reconstruction and Interaction Using a Moving Depth Camera. In *Proc. of the ACM Symp. on User Interface Software and Technology*, pp. 559–568, 2011.

[10] B. Jones et al. RoomAlive: Magical Experiences Enabled by Scalable, Adaptive Projector-camera Units. In *Proc. of the Annual Symp. on User Interface Software and Technology*, pp. 637–644, 2014.

[11] 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*, pp. 500–516, 2016.

[12] 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. *IEEE Trans. on Visualization and Computer Graphics*, 21(11):1241–1250, 2015.

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

[14] M. Keller, D. Lefloch, M. Lambers, S. Izadi, T. Weyrich, and A. Kolb. Real-Time 3D Reconstruction in Dynamic Scenes Using Point-Based Fusion. In *Proc. of Joint 3DIM/3DPVT Conference*, p. 8, 2013.

[15] W. E. Lorensen and H. E. Cline. Marching Cubes: A High Resolution 3D Surface Construction Algorithm. In *Proc. of the 14th Annual Conf. on Computer Graphics and Interactive Techniques*, pp. 163–169, 1987.

[16] R. Maier, R. Schaller, and D. Cremers. Efficient Online Surface Correction for Real-time Large-Scale 3D Reconstruction. In *British Machine Vision Conference (BMVC)*, 2017.

[17] A. Maimone, J. Bidwell, K. Peng, and H. Fuchs. Enhanced personal autostereoscopic telepresence system using commodity depth cameras. *Computers & Graphics*, 36(7):791 – 807, 2012.

[18] A. Maimone and H. Fuchs. Real-time volumetric 3D capture of room-sized scenes for telepresence. In *Proc. of the 3DTV-Conference*, 2012.

[19] D. Molyneaux, S. Izadi, D. Kim, O. Hilliges, S. Hodges, X. Cao, A. Butler, and H. Gellersen. Interactive Environment-Aware Handheld Projectors for Pervasive Computing Spaces. In *Proc. of the Int. Conf. on Pervasive Computing*, pp. 197–215, 2012.

[20] A. Mossel and M. Kröter. Streaming and exploration of dynamically changing dense 3d reconstructions in immersive virtual reality. In *Proc. of IEEE Int. Symp. on Mixed and Augmented Reality*, pp. 43–48, 2016.

[21] R. A. Newcombe et al. KinectFusion: Real-Time Dense Surface Mapping and Tracking. In *Proc. of IEEE Int. Symp. on Mixed and Augmented Reality*. IEEE, 2011.

[22] R. A. Newcombe, D. Fox, and S. M. Seitz. DynamicFusion: Reconstruction and tracking of non-rigid scenes in real-time. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 343–352, 2015.

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

[24] S. Orts-Escolano et al. Holoportation: Virtual 3D Teleportation in Real-time. In *Proc. of the Annual Symp. on User Interface Software and Technology*, pp. 741–754, 2016.

[25] F. Reichl, J. Weiss, and R. Westermann. Memory-Efficient Interactive Online Reconstruction From Depth Image Streams. *Computer Graphics Forum*, 35(8):108–119, 2016.

[26] H. Roth and M. Vona. Moving volume kinectfusion. In *Proc. of the British Machine Vision Conference*, pp. 112.1–112.11, 2012.

[27] P. Stotko, S. Krumpen, M. B. Hullin, M. Weinmann, and R. Klein. SLAMCast: Large-Scale, Real-Time 3D Reconstruction and Streaming for Immersive Multi-Client Live Telepresence. *IEEE Trans. on Visualization and Computer Graphics*, 25(5):2102–2112, 2019.

[28] R. Vasudevan, G. Kurillo, E. Lobaton, T. Bernardin, O. Kreylos, R. Bajcsy, and K. Nahrstedt. High-Quality Visualization for Geographically Distributed 3-D Teleimmersive Applications. *IEEE Trans. on Multimedia*, 13(3):573–584, 2011.

[29] T. Whelan, H. Johannsson, M. Kaess, J. J. Leonard, and J. McDonald. Robust Real-Time Visual Odometry for Dense RGB-D Mapping. In *IEEE Int. Conf. on Robotics and Automation*, pp. 5724–5731, 2013.

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

[31] 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 Int. Journal of Robotics Research*, 34(4-5):598–626, 2015.

[32] Q.-Y. Zhou and V. Koltun. Dense Scene Reconstruction with Points of Interest. *ACM Trans. Graph.*, 32(4):112, 2013.

Publication:
"Albedo estimation for real-time 3D reconstruction using RGB-D and IR data"

Patrick Stotko, Michael Weinmann, and Reinhard Klein

ISPRS Journal of Photogrammetry and Remote Sensing (P&RS)

2019

DOI: 10.1016/j.isprsjprs.2019.01.018

ISPRS Journal of Photogrammetry and Remote Sensing 150 (2019) 213–225

Contents lists available at ScienceDirect

ISPRS Journal of Photogrammetry and Remote Sensing

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

Albedo estimation for real-time 3D reconstruction using RGB-D and IR data

Patrick Stotko*, Michael Weinmann, Reinhard Klein

Institute of Computer Science II – Computer Graphics, University of Bonn, Endenicher Allee 19a, 53115 Bonn, Germany

ARTICLE INFO

Keywords:
Real-time
Reconstruction
Textures
Reflectance
Image segmentation
Infrared

ABSTRACT

Reconstructing scenes in real-time using low-cost sensors has gained increasing attention in recent research and enabled numerous applications in graphics, vision, and robotics. While current techniques offer a substantial improvement regarding the quality of the reconstructed geometry, the degree of realism of the overall appearance is still lacking as the reconstruction of accurate surface appearance is highly challenging due to the complex interplay of surface geometry, reflectance properties and surrounding illumination. We present a novel approach that allows the reconstruction of both the geometry and the spatially varying surface albedo of a scene from RGB-D and IR data obtained via commodity sensors. In comparison to previous approaches, our approach offers an improved robustness and a significant speed-up to even fulfill the real-time requirements. For this purpose, we exploit the benefits of scene segmentation to improve albedo estimation due to the resulting better segment-wise coupling of IR and RGB data that takes into account the wavelength characteristics of different materials within the scene. The estimated albedo is directly integrated into the dense volumetric reconstruction framework using a novel weighting scheme to generate high-quality results. In our evaluation, we demonstrate that our approach allows albedo capturing of complicated scenarios including complex, high-frequent and strongly varying lighting as well as shadows.

1. Introduction

Due to the rapidly spreading availability of affordable RGB-D sensors included in capturing devices like the Microsoft Kinect or recent mobile devices and the increasing computational power of GPUs, real-time 3D reconstruction has gained a lot of attention in recent years and enabled numerous applications in graphics, vision and robotics. Inferring accurate 3D models in terms of both geometry and material characteristics is of great importance to enable a better immersive experience of objects when inspecting or interacting with a captured scene. For instance, the fast digitization of scenes also receives a lot of interest in architecture and entertainment applications where the digitized model may be used with exchanged illumination conditions as given for different day/night times or different weather conditions. When looking at a certain scene, we perceive the inherent interplay of the geometric structure and the reflectance behavior of the surfaces in the considered scene as well as the present illumination conditions that results in the overall scene appearance. Therefore, incrementally captured color observations may vary significantly for a particular surface point due to view-dependent and illumination-dependent shading effects such as shadows and high-frequency lighting characteristics. However, decoupling the surface albedo from these environment-specific characteristics as required by applications where the scene has to be depicted in a manipulated environment such as different lighting conditions, is highly ill-posed and non-trivial. In addition, this task becomes even more challenging due to the high demands regarding efficiency imposed on real-time reconstruction approaches. With this paper, we directly address these issues with an intrinsic image decomposition framework that is based on an efficient optimization to allow real-time performance. In addition to the significantly faster inference of albedo and illumination characteristics in comparison to previous work, our approach also offers an improved robustness which becomes particularly evident for scenes with complex, high-frequent and strongly varying illumination.

Since the introduction of the seminal KinectFusion real-time reconstruction framework (Izadi et al., 2011; Newcombe et al., 2011), much effort has been spent on improving run-time and reconstruction quality (Nießner et al., 2013; Chen et al., 2013; Whelan et al., 2012; Kähler et al., 2016b; Dai et al., 2017). However, these approaches are still lacking regarding the degree of surface texture quality in their reconstructions induced by an insufficient scene representation (see Figs. 1 and 10). As a result of fusing view-dependent and illumination-dependent color images into a texture, shading effects such as shadows and highlights are not separated and the appearance of the

* Corresponding author.
E-mail addresses: stotko@cs.uni-bonn.de (P. Stotko), mw@cs.uni-bonn.de (M. Weinmann), rk@cs.uni-bonn.de (R. Klein).

https://doi.org/10.1016/j.isprsjprs.2019.01.018
Received 10 August 2018; Received in revised form 21 November 2018; Accepted 21 January 2019

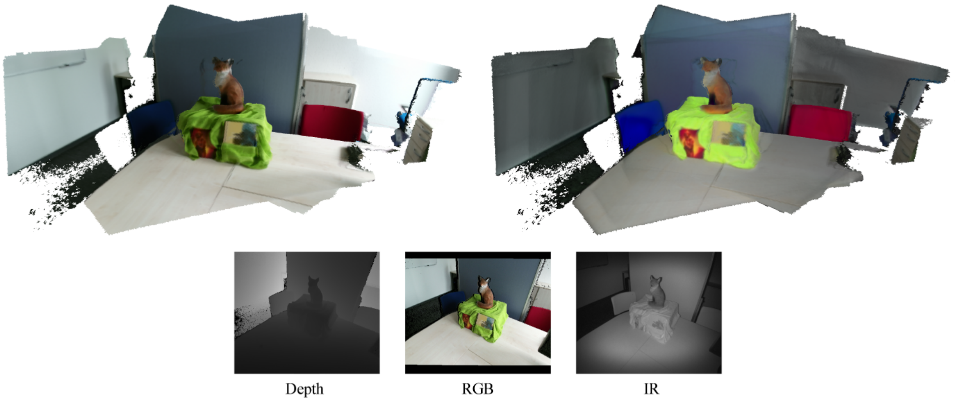


Fig. 1. We present a novel approach that allows the automatic real-time reconstruction of both geometry and reflectance information in terms of spatially varying surface albedo of static scenes. For the input depth, RGB and IR data (illustrated in the bottom row for an exemplary image within the sequence) the reflectance layer estimated using our technique (right) is more accurate than the one obtained with previous approaches where viewpoint and illumination dependent effects such as shadows are stored within the mapped texture information (left). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

reconstructed model exhibits inconsistencies when analyzing the scene from different viewpoints or performing scene relighting. This, in turn, may even lead to wrong impressions regarding the corresponding material properties. Among the few approaches that consider on-line color texture reconstruction, Whelan et al. (2015) improve texture consistency by rejecting samples at object boundaries or grazing angles. However, shadows cannot be handled in this way since the appearance of an object is much more complex and depends on the complex interplay of surface geometry, surface reflectance properties, and the surrounding illumination conditions in the scene. Towards a more accurate scene reconstruction regarding reflectance properties – which is well-known to be a severely ill-posed and hard-to-solve task, particularly under the requirement of real-time performance – Meka et al. (2017) incorporate user-provided constraints to estimate the surface albedo and geometry interactively, however, only in a semi-automatic way. Furthermore, Kerl et al. (2014) propose an additional constraint using the IR data provided by the Kinect v2 to improve the decomposition into a reflectance and shading term in scenarios with complex high-frequent illumination.

In this paper, we propose a novel practical automatic large-scale reconstruction technique that jointly estimates geometry and surface appearance in terms of spatially-varying albedo in real-time. For this purpose, we leverage the IR data provided by the Kinect v2 to improve the robustness of the albedo estimation similar to the approach by Kerl et al. (2014) but use a more general approach based on a soft scene segmentation. Thereby, our technique allows a more flexible decomposition into an albedo and shading term and overcomes the limitation of the approach by Kerl et al. (2014) regarding the implicit assumption that for all materials in the scene there is only one proportionality factor between the IR channel and the RGB channels which is violated in real scenarios (see Fig. 3). In addition, we apply a new data propagation technique that greatly improves the performance of the whole reflectance estimation pipeline. Inspired by a state-of-the-art volumetric reconstruction pipeline, we densely fuse the acquired geometry and surface reflectance information. Furthermore, efficiency is gained by the use of a novel Total Variation (TV) solver that is particularly designed to handle high-framerate data. By exploiting frame-to-frame coherency for initializing the optimization, a speed-up of up to a factor of 40 compared to traditional approaches makes the approach run in real-time.

In summary, the main contributions of this paper are:

- A novel fully-automatic technique to reconstruct surface geometry together with reflectance information in terms of albedo information at real-time rates and with the texture resolution of state-of-the-art real-time reconstruction frameworks (voxel resolution) based on integrating intrinsic image decomposition into a real-time scene reconstruction framework.
- A novel approach for the robust estimation of surface reflectance properties from RGB-D and IR data based on soft scene segmentation.
- A dedicated total variation solver that exploits the high frame-rates of the Kinect and improves the decomposition run-time performance by more than one order of magnitude.

2. Related work

3D Reconstruction. Real-time scene reconstruction has been a challenging topic for decades and rapidly gained increasing interest with the success of the KinectFusion system (Izadi et al., 2011; Newcombe et al., 2011). Based on a volumetric fusion principle (Curless and Levoy, 1996), the first automated real-time reconstruction of indoor scenes was achieved using a low-cost Kinect sensor. Unfortunately, the compelling results were achieved at the cost of an extremely limited size of the working volume due to the high GPU memory requirements. Recent developments tried to relax this by using moving volume techniques (Whelan et al., 2012; Whelan et al., 2015), hierarchical data structures (Chen et al., 2013), and sparse voxel block hashing (Nießner et al., 2013; Kähler et al., 2015, 2016a). Another drawback of KinectFusion limiting the reconstruction quality was the accuracy of the used camera pose estimation algorithm. In the meantime, several registration approaches have been proposed to increase the robustness of the estimated poses (Stotko, 2016). Several loop closure techniques (Kähler et al., 2016b; Dai et al., 2017) have further improved the results by enforcing global consistency constraints. Very recently, Liu et al. (2018) used a stability-based sampling method and exploited additional IMU data to improve camera tracking accuracy. Rajput et al. (2018) applied depth outlier removal and denoising based on total variation for improving the accuracy regarding tracking and 3D geometry. A more comprehensive survey in the context of 3D reconstruction can be found in the state-of-the-art report by Zollhöfer et al. (2018).

Color texture reconstruction. The aforementioned approaches mainly focus on the quality of the reconstructed geometry and only consider the averaging of RGB values to obtain the texture and spend little focus on enforcing global texture consistency. Whelan et al. (2015) rejected samples at object borders or grazing viewing angles that would lead to undesirable updates. In subsequent work, they estimated the light source positions in the scene and rejected samples containing specular highlights (Whelan et al., 2016). Li et al. (2016) further improved the reconstructed textures by estimating the varying exposure time of the Kinect and correcting the brightness of the input color images. In the context of off-line reconstruction, Zhou and Koltun (2014) proposed a system that jointly estimates camera poses and vertex colors by maximizing photometric consistency. Bi et al. (2017) built upon this work and used patch-based segmentation to define patch-based consistency constraints. Optimization of camera poses, geometry and surface albedo in the volumetric domain has also been considered to obtain high quality color textures (Maier et al., 2017).

Intrinsic image decomposition. The main drawback of only reconstructing color textures lies in the fact that, although highlights and

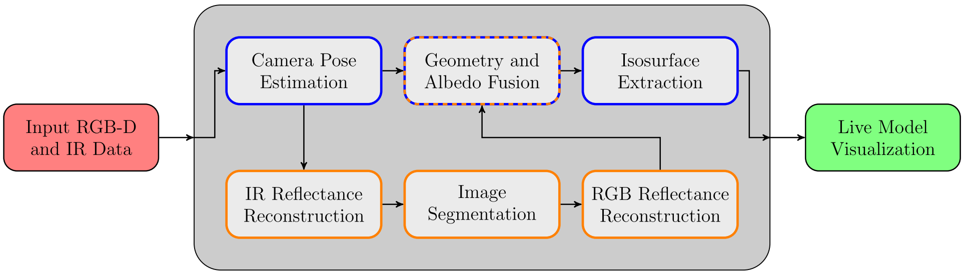


Fig. 2. Our novel appearance reconstruction technique fully-automatically estimates and fuses geometry and albedo information in real-time. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

exposure artifacts can be factored out, shadows and other illumination effects are still fused into the texture. Instead, intrinsic image decomposition approaches overcome this problem by factoring the observed color image into an reflectance image, which contains the surface-specific diffuse albedo information, and a shading image containing all illumination-dependent effects. To disambiguate the inherently ambiguous intrinsic image decomposition problem, several approaches assume distant low-frequent illumination modeled by first or second order spherical harmonics (Wu et al., 2014; Barron and Malik, 2013, 2015). However, in case of sunlight shining through a window or non-distant light sources, strong shading variations cannot be handled and will be propagated into the reflectance layer instead of the shading layer where they should occur. Shi et al. (2015) used super-pixel clustering to group regions with similar reflectance behavior to improve run-time performance by reducing the number of unknowns and implicitly enforce a reflectance constancy prior. While this approach works for (textured) objects consisting of a set of uniform albedo components, texture gradients as well as overlapping colors and reflectance variations cannot be handled and are left to the shading map. Similarly, Jin and Gu (2017) combined super-pixel clustering and intrinsic image decomposition for hyperspectral images. In contrast, Meka et al. (2016) developed a real-time approach that solves the intrinsic image decomposition problem automatically on the GPU for video data. In subsequent work, they integrated their approach into a volumetric reconstruction framework and incorporated user-provided constraints which stabilize the results and allow interactive applications like material editing and recoloring (Meka et al., 2017). Kerl et al. (2014) exploited the infrared data of the Kinect v2 to disambiguate the intrinsic image decomposition problem by coupling RGB and IR observations. Since the exposure time of the IR camera is fixed in comparison to the RGB camera, coupling also implicitly enforces temporal consistency when video data is considered. In the context of dynamic scene reconstruction, the real-time approach by Guo et al. (2017) is tailored to single moving objects. For a more comprehensive survey in the context of reflectance estimation from intrinsic image decomposition, we refer to the state-of-the-art report by Bonneel et al. (2017).

While we built upon the work of Kerl et al. (2014), however, in contrast to this approach, our approach benefits from a better coupling of the IR albedo and the RGB albedo based on a segment-wise consideration. This allows to overcome the implicit assumption that for all materials in the scene there is only one proportionality factor between the IR channel and the RGB channel which is violated in real scenarios. While the incorporation of segmentation into image decomposition is not new in this context (Shi et al., 2015; Jin and Gu, 2017), we, to the best of our knowledge, for the first time use a soft segmentation for RGB-D and IR data. By using soft segmentation instead of hard segmentation, we avoid artifacts like overlapping reflectance colors or reflectance variations in super-pixels as mentioned by Shi et al. (2015). Furthermore, we exploit frame-to-frame coherency for improved decomposition performance and use standard 3D reconstruction frameworks (Nießner et al., 2013; Kähler et al., 2015) to jointly fuse geometry and estimated albedo observations into a consistent global 3D model.

Surface appearance and illumination reconstruction. While our approach focuses on real-time surface albedo and geometry recovery, some techniques tried to reconstruct surface geometry together with spatially varying surface reflectance characteristics and surrounding environment illumination. Wu and Zhou (2015) introduced the AppFusion framework which, using a mirror ball to capture the surrounding environment illumination as well as exploiting RGB-D and IR data provided by the involved sensor, reconstructs the spatially varying diffuse and specular albedo of an object along with its geometry. However, the region of interest was limited by the markers involved for camera tracking/registration and the algorithm required multiple capturing passes and manual refinement by the user. Recently, further approaches have been proposed that, however, all are lacking regarding real-time capability (Hachama et al., 2015; Richter-Trummer et al., 2016; Wu et al., 2016; Zuo et al., 2017). Very recently, Meka et al. (2018) applied deep learning techniques to estimate the surface reflectance of a single object with uniform appearance.

3. Overview

As illustrated in Fig. 2, our proposed reconstruction pipeline fully-automatically processes incoming RGB-D and IR data streams as obtained by commodity sensors such as the Kinect v2 and infers a 3D model with attached albedo information in real-time. This is achieved based on an architecture that can be divided into a geometry-related part (blue-framed steps) inspired by a state-of-the-art 3D volumetric reconstruction approach (Nießner et al., 2013) and a reflectance-related part (orange-framed steps).

First, the camera pose is estimated to get a transformation mapping from the current local camera coordinate system to the global one. The camera pose as well as the RGB-D and IR data are then used to reconstruct the IR Shading Model consisting of an ambient term and a diffuse albedo map (Section 5.1). Using the input color image and the estimated infrared albedo map, the image space is clustered into a set of segments (Section 5.3). Our novel intrinsic image decomposition formulation incorporates these information to robustly reconstruct the current diffuse color albedo image (Section 5.2). Finally, the geometry fusion step of the 3D reconstruction framework is extended to also fuse the estimated albedo information robustly using a confidence-based weighting scheme (Section 6). This is followed by the extraction of the current isosurface of the stored volume to provide the user live feedback of the capturing process. Furthermore, we improve the run-time of these approaches using a dedicated total variation solver that exploits the high frame-rates of the Kinect (Sections 7.1 and 7.2). Details of these involved components will be provided in the following sections.

4. Sensor

Estimating the albedo solely from the RGB data is a highly-challenging task since the illumination of the scene may be arbitrarily complex and may result in large variations of surface appearance. In contrast, active scanning devices performing measurements in the infrared (IR) domain such as the Kinect v2 allow for a more robust modeling of the observed radiance in this domain due to knowledge regarding the light source position and characteristics. Therefore, we take advantage of the additional depth and IR data of the Kinect v2 camera to improve the robustness of the albedo estimation approach.

The sensor uses time-of-flight (ToF) technology to estimate the distance to the objects in the scene. The resulting measurement is a depth image with values within a range between approximately 0.5 m and 4.5 m (Payne et al., 2014; Pagliari and Pinto, 2015). Here, modulated near infrared (NIR) light with a fixed wavelength of 860 nm and a modulation frequency between 10 MHz and 130 MHz (Payne et al., 2014; Valgma, 2016) is sent out by the emitter, travels through the scene and is reflected back to the sensor. The object distance is estimated from the phase of the reflected light and the time until it is captured by the IR camera. Since the depth range and emitter intensity are known and fixed, the IR camera uses a fixed exposure time for capturing temporally coherent IR images with 11-bit dynamic range.

In the fields of robotics, depth refinement and others, the reliability of the IR and depth data provided by the Kinect v2 was evaluated in various indoor and outdoor scenarios. The effect of the natural illumination in indoor scenes and outdoor overcast situations is negligible and is dominated by the emitter's signal (Zennaro et al., 2015; Fankhauser et al., 2015). Choe et al. (2014, 2017) demonstrated similar results even with the presence of a wide spectrum light source. For sun light directly facing the sensor, large amounts of the IR and depth data are not reliable anymore but fortunately classified as invalid by the sensor and rejected (Fankhauser et al., 2015). Therefore, similar to previous work (Kerl et al., 2014; Wu and Zhou, 2015; Wu et al., 2016; Guo et al., 2017; Meka et al., 2017), we assume that the camera is calibrated and artifacts caused by effects such as multi-path interference and misalignments between RGB and depth/IR images are not introduced in both the geometry and appearance reconstruction.

5. Albedo estimation

We build our approach on top of an initial estimation of the reflectance properties in the infrared channel and use these information for the computation of the RGB albedo which we additionally stabilize based on a prior segmentation step. In the following, we will discuss the individual albedo estimation steps in detail. An in-depth explanation of the used notation can be found in Table 1.

5.1. IR reflectance reconstruction

We use the physically motivated definition of light transport to model the reflectance reconstruction problem. For a point $\boldsymbol{x}$, the observed radiance $\boldsymbol{L}_o$ that is reflected into direction $\boldsymbol{v}$ is defined by the Rendering Equation (Kajiya, 1986)

$$\boldsymbol{L}_o(\boldsymbol{x}, \boldsymbol{v}) = \boldsymbol{L}_e(\boldsymbol{x}, \boldsymbol{v}) + \int_{\mathcal{H}} \boldsymbol{f}_{BRDF}(\boldsymbol{l}_i, \boldsymbol{x}, \boldsymbol{v}) \boldsymbol{L}_i(\boldsymbol{x}, \boldsymbol{l}_i) < \boldsymbol{l} \,|\, \boldsymbol{n} > d\omega_i \tag{1}$$

Table 1
Symbols and notation used throughout this paper.

Symbol	Explanation
$\boldsymbol{p}$	Image pixel in $[0, w) \times [0, h)$
$\boldsymbol{x} \in \mathbb{R}^3$	Surface point
$\boldsymbol{n} \in \mathbb{R}^3$	Surface normal at point $\boldsymbol{x}$
$r \in \mathbb{R}$	Distance of point $\boldsymbol{x}$to the infrared emitter
$\boldsymbol{l} \in \mathbb{R}^3$	Incoming light direction
$\boldsymbol{v} \in \mathbb{R}^3$	View direction to camera
L_{IR}	Input infrared image
$\varkappa_{d,IR}$	Diffuse infrared albedo image
$L_{a,IR}$	Ambient infrared radiance value
I_{IR}	Intensity of infrared emitter
$\boldsymbol{L}_{RGB}$	Input RGB radiance image
$\boldsymbol{\varkappa}_{d,RGB}$	Diffuse RGB albedo image
s_{RGB}	RGB shading image

where $\boldsymbol{f}_{BRDF}$ is the light- and view-dependent reflectance, $\boldsymbol{L}_i$ the incident radiance from direction $\boldsymbol{l}_i$, and $\boldsymbol{L}_e$ the radiance if the surface itself is emitting light, i.e. is a light source. In our IR scenario, we assume that the objects are predominantly diffuse and that the sensor is the only/dominant light source in the scene. In contrast to Kerl et al. (2014), we model the observed radiance in the IR image L_{IR}, where vignetting effects at the borders are corrected using a pre-computed mask, similar to Or-El et al. (2016) by an indirect and a direct illumination term according to

$$L_{IR}(\boldsymbol{p}) = \varkappa_{d,IR}(\boldsymbol{p})\cdot L_{a,IR} + \frac{\varkappa_{d,IR}(\boldsymbol{p})}{\pi}\cdot < \boldsymbol{l} \,|\, \boldsymbol{n} > \frac{I_{IR}}{r^2} \tag{2}$$

where $\varkappa_{d,IR}$ denotes the diffuse albedo that has to be determined, $L_{a,IR}$ the ambient radiance, I_{IR} the known intensity of the IR emitter, and $\boldsymbol{n}$ the normal of the surface point seen at pixel $\boldsymbol{p}$. Therefore, the surface reflectance reduces to a diffuse term $\frac{\varkappa_{d,IR}(\boldsymbol{p})}{\pi}$ (Lambert, 1760; Guarnera et al., 2016) and the direct radiance $\frac{I_{IR}}{r^2}$ is proportional to the emitter's intensity. For the ambient part, we approximate and summarize all contributing factors into the ambient radiance $L_{a,IR}$. Although the normals are computed from noisy input depth images, slight inaccuracies in the normal directions do not affect the reconstruction quality of the color albedo map since the introduced noise is regularized using a total variation prior. We approximate the incoming light direction $\boldsymbol{l}$ by the viewing direction $\boldsymbol{v}$ since emitter and camera are only a few centimeters apart in the sensor. As a consequence, the distance $\boldsymbol{r}$ between the surface point $\boldsymbol{x}$ that is observed at pixel $\boldsymbol{p}$ can be computed as the distance of $\boldsymbol{x}$ to the camera.

Inspired by the approach of Or-El et al. (2016), we iteratively alternate the optimization for the ambient radiance $L_{a,IR}$ and the diffuse albedo $\varkappa_{d,IR}$ while keeping the other variable fixed. First, we define an ambient residual image $\Delta L_{a,IR}$ containing only the shading effects caused by the ambient term and minimize the following least-squares energy

$$E_{a,IR}(L_{a,IR}) = \sum_{\boldsymbol{p}} \|\Delta L_{a,IR}(\boldsymbol{p}) - \varkappa_{d,IR}(\boldsymbol{p})\cdot L_{a,IR}\|_2^2 \tag{3}$$

to obtain an estimate of the ambient radiance. Subsequently, we derive an estimate of the diffuse albedo map $\varkappa_{d,IR}$ by the optimization of the energy

$$\begin{aligned} E_{d,IR}(\varkappa_{d,IR}) = \lambda_{d,IR} \left\| L_{IR} - \varkappa_{d,IR}\cdot\left(L_{a,IR} + \frac{\langle \boldsymbol{l} \mid \boldsymbol{n} \rangle}{\pi}\frac{I_{IR}}{r^2}\right) \right\|_2^2 \\ + \|w_{d,IR}\cdot\nabla\varkappa_{d,IR}\|_1 \end{aligned} \tag{4}$$

consisting of a least-squares data term that penalizes deviations from the infrared shading model and a weighted L1 total variation smoothness term. For the sake of simplicity and readability, we only mention the pixel variable $\boldsymbol{p}$ when needed and define optimization problems for images pixel-wise to solve them in parallel on the GPU (see Section 7). Here, the scalar $\lambda_{d,IR}$ controls the trade-off between fidelity and smoothness of the solution. In contrast to Or-El et al. (2016), we do not consider the albedo image gradient with respect to a manifold. Instead, we weight each gradient value by

$$w_{d,IR}(\boldsymbol{p}) = \exp\left(-\frac{\|\nabla \boldsymbol{n}(\boldsymbol{p})\|_2}{\sigma_{d,IR}}\right) \tag{5}$$

to further guide the optimizer to the desired solution. The weight values describe a local curvature-like measure based on the gradient of the normal map $\boldsymbol{n}$ and ensure sharp edges in the albedo map. Typically, changes of the albedo are observed at object boundaries where the local curvature is high.

5.2. RGB reflectance reconstruction

We use the inferred infrared model to solve the highly ambiguous intrinsic image decomposition problem

$$\boldsymbol{L}_{RGB}(\boldsymbol{p}) = \kappa_{d,RGB}(\boldsymbol{p})\cdot s_{RGB}(\boldsymbol{p}) \tag{6}$$

where $\boldsymbol{L}_{RGB}$ denotes the observed RGB radiance image, $\boldsymbol{\kappa}_{d,RGB}$ the diffuse color albedo map and s_{RGB} the shading map. Like most other decomposition approaches, we assume that the scene is predominantly diffuse and illuminated with white light, which leads to scalar-valued shading images. Furthermore, we assume that the radiance and albedo values lie in the unit interval and that the shading values are non-negative. Whereas the radiance image L_{IR}, the ambient radiance $L_{a,IR}$ and the diffuse albedo image $\kappa_{d,IR}$ in the infrared model are all defined in linear space, their counterparts in the RGB model are given in gamma space. In this way, equal differences in the energy are also observed perceptually equal. To derive an estimate for the unknown albedo and shading images $\boldsymbol{\kappa}_{d,RGB}$ and s_{RGB}, we propose the following energy functional

$$\begin{aligned} E_{RGB}(\boldsymbol{\kappa}_{d,RGB}, s_{RGB}) = {} & E_{data,RGB}(\boldsymbol{\kappa}_{d,RGB}, s_{RGB}) \\ & + E_{coup,RGB}(\boldsymbol{\kappa}_{d,RGB}) \\ & + E_{reg,RGB}(\boldsymbol{\kappa}_{d,RGB}, s_{RGB}) \end{aligned} \tag{7}$$

consisting of a data term $E_{data,RGB}$, a coupling term $E_{coup,RGB}$, and a regularization term $E_{reg,RGB}$.

Data term. Similar to previous approaches (Meka et al., 2016, 2017), we constrain the solution to fulfill the intrinsic image decomposition, i.e. we penalize

$$E_{data,RGB}(\boldsymbol{\kappa}_{d,RGB}, s_{RGB}) = \lambda_{data,RGB}\ \|\boldsymbol{L}_{RGB} - \boldsymbol{\kappa}_{d,RGB}\cdot s_{RGB}\|_2^2 \tag{8}$$

Here, the parameter $\lambda_{data,RGB}$ controls the influence of this soft constraint.

Coupling term. In order to disambiguate the intrinsic image decomposition problem and get a unique temporal consistent solution even for video data, we couple the diffuse color albedo with its infrared version that has been estimated before. Kerl et al. (2014) performed this coupling globally for the complete image using the term

$$\|\kappa_{d,IR} - \boldsymbol{g}\cdot\boldsymbol{\kappa}_{d,RGB}\|_2^2 \tag{9}$$

where the factor $\boldsymbol{g}$ is chosen such that the mean albedo equals the mean color value:

$$\boldsymbol{g} = \frac{1}{3}\frac{\bar{\kappa}_{d,IR}}{\bar{\boldsymbol{L}}_{RGB}^{\top}} \in \mathbb{R}^{1\times 3} \tag{10}$$

As demonstrated in Fig. 3, such a coupling across the whole image leads to undesirable results as the reflectance may change widely across different wavelengths for most materials. Thus, the ratio between color and infrared albedo values can be large. Image-wide coupling not only shifts the mean albedo towards the mean radiance, but also the albedo ratio towards the mean ratio, i.e. the coupling factor $\boldsymbol{g}$.

To overcome this problem, we divide the image into a set of k segments and compute the coupling factor per segment rather than per image:

$$E_{coup,RGB}(\boldsymbol{\kappa}_{d,RGB}) = \lambda_{coup,RGB}\sum_{i=1}^{k} v_i\ \|\kappa_{d,IR} - \boldsymbol{g}_i\cdot\boldsymbol{\kappa}_{d,RGB}\|_2^2 \tag{11}$$

Here, v_i denotes the probability map that associates each pixel the probability to be coupled with the i-th segment. This way, the gap between albedo ratio and the segment ratio is much smaller.

Regularization term. Like in the infrared shading model described in Section 5.1, we use a smoothness prior for the color albedo image. Kerl et al. (2014) also added such a prior for the shading image and alternate the optimization between both images in order to make the problem feasible. In contrast, we observed that enforcing smoothness only for the albedo image is already sufficient and leads to almost identical results as the data term implicitly handles this constraint. Furthermore, sharp edges in the radiance image caused by shading effects would directly be propagated to the shading image and not accidentally get smoothed out. Instead of spending additional time for an alternating optimization of albedo and shading, we prefer to use the saved time for the reconstruction in the infrared model to get a more accurate albedo there. As a consequence, only the data term depends on the shading image which leads to a local least-squares solution per pixel. Unfortunately, the convergence speed decreases as the data term error immediately gets very small. We overcome this issue by adding a term that penalizes the temporal gradient of the shading image during the optimization, i.e. we estimate the shading map s_{RGB} by optimizing the energy

$$\begin{aligned} E_{reg,RGB}(\boldsymbol{\kappa}_{d,RGB}, s_{RGB}) = {} & \lambda_{reg,RGB}\ \|s_{RGB} - s_{RGB}^{(j-1)}\|_2^2 \\ & + \|w_{d,RGB}\cdot\nabla\boldsymbol{\kappa}_{d,RGB}\|_1 \end{aligned} \tag{12}$$

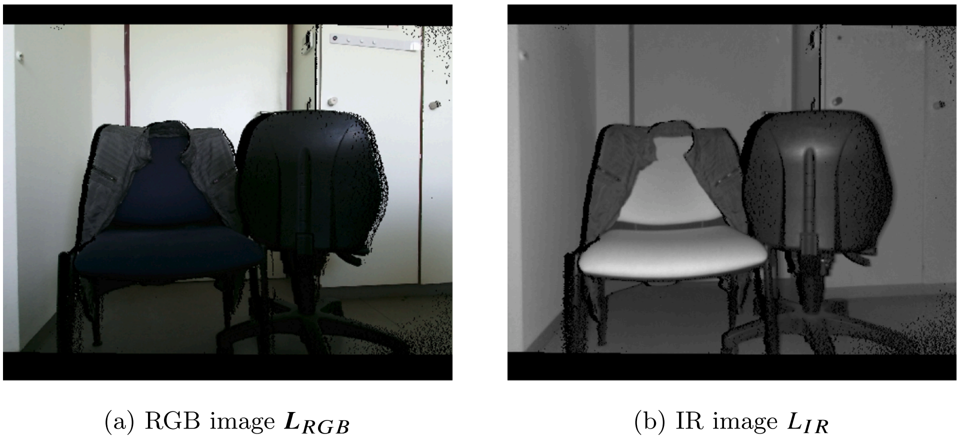
Here, $s_{RGB}^{(j-1)}$ is the shading image from the previous iteration $j-1$ and $\lambda_{reg,RGB}$ controls the strength of the dampening. In addition, we weight the color albedo gradients based on the infrared albedo image gradient according to

$$w_{d,RGB}(\boldsymbol{p}) = \exp\left(-\frac{\|\nabla\kappa_{d,IR}(\boldsymbol{p})\|_2}{\sigma_{d,RGB}}\right) \tag{13}$$

Edges in the color albedo image are likely if there is also an edge in the infrared albedo image.

5.3. Image segmentation

As mentioned earlier, we consider a segment-wise computation of the coupling factors for the coupling energy term to improve the robustness of the albedo estimation algorithm. In our case, segments


(a) RGB image $\boldsymbol{L}_{RGB}$

(b) IR image L_{IR}

Fig. 3. Observed radiance may change drastically for some materials like cloth. Compared to black plastic (right chair) and the gray jacket, the radiance of the dark blue cloth cover (left chair) is much larger in the infrared channel. Black pixels correspond to missing or unreliable data in the RGB or depth/IR image (see Section 4). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

should also follow brightness variations as obtained for shadowed areas, which is in contrast to semantic segmentation where the segments should follow semantic entities. Since segmentation is a challenging task itself, we apply different strategies and evaluate their performance in the scope of our framework.

Color-based hard clustering. Given the color radiance image $\boldsymbol{L}_{RGB}$ converted to the LAB color space and the diffuse infrared albedo map $\kappa_{d,IR}$, we define the feature image $\boldsymbol{f}$ to be composed of the two LAB color components and the diffuse infrared albedo. We normalize the LAB components to the unit interval to match the range of the infrared albedo values. Assuming that the scene can be divided into k segments, we estimate the probability $v_i(\boldsymbol{p})$ that the pixel $\boldsymbol{p}$ belongs to the i-th segment. In our segmentation energy of the form

$$E_{hard,seg}(v_i, \boldsymbol{c}_i) = \lambda_{seg} \sum_{i=1}^{k} \left(\sum_{\boldsymbol{p}} v_i(\boldsymbol{p}) \|\boldsymbol{f}(\boldsymbol{p}) - \boldsymbol{c}_i\|_2^2 \right) + \|w_{seg} \cdot \nabla \boldsymbol{v}\|_1 \tag{14}$$

this is modeled by penalizing the squared distance of the data value $\boldsymbol{f}(\boldsymbol{p})$ to the cluster center $\boldsymbol{c}_i$ weighted by the probability $v_i(\boldsymbol{p})$. In addition, we enforce that the segmentation is smooth in the sense of the L1 total variation. The regularization weights

$$w_{seg}(\boldsymbol{p}) = \exp\left(-\frac{\|\nabla \boldsymbol{n}(\boldsymbol{p})\|_2}{\sigma_{seg}} \right) \tag{15}$$

depend on the curvature such as in Section 5.1 to enforce that the segments are aligned with the object boundaries. Finally, we obtain the final clustering by choosing the segment with the highest probability for each pixel.

Color-based soft clustering. In addition to the previously mentioned hard clustering strategy, we also consider soft clustering as an alternative technique to separate materials with different reflectance properties. For this purpose, we use a slightly modified energy formulation that penalizes perceptual differences of colors:

$$E_{soft,seg}(v_i, \boldsymbol{c}_i) = \lambda_{seg} \sum_{i=1}^{k} \left(\sum_{\boldsymbol{p}} v_i(\boldsymbol{p}) \|\boldsymbol{e}(\boldsymbol{L}_{RGB}(\boldsymbol{p})) - \boldsymbol{e}(\boldsymbol{c}_i \cdot \kappa_{d,IR}(\boldsymbol{p}))\|_2^2 \right) + \|w_{seg} \cdot \nabla \boldsymbol{v}\|_1 \tag{16}$$

Here, $\boldsymbol{e}$ denotes the transformation from the RGB to the LAB color space. The cluster centers $\boldsymbol{c}_i$ couple the color and infrared domain similar to the factors used for color albedo reconstruction.

Geometry-based hard clustering. Another strategy is the segmentation of a scene into a set of objects based on the observed geometry since material properties change at object boundaries. Therefore, we use the approach by Tateno et al. (2016) to reconstruct a model segmentation and compare this to the color-based approach described above.

6. Geometry and albedo fusion

After the computation of the camera pose and the albedo maps, we have to fuse the geometry information given as depth maps provided by the sensor and the estimated albedo map to a consistent scene model. For this purpose, we developed a fusion pipeline inspired by the VoxelHashing technique by Nießner et al. (2013). Like in the KinectFusion framework (Izadi et al., 2011; Newcombe et al., 2011), the 3D reconstruction is stored implicitly using a discretized truncated signed distance field (TSDF). Each voxel in the volume stores a TSDF value together with a weight needed for the update and a color value. The captured depth data $z^{(t+1)}$ from the new time step $t + 1$ are fused into the volume based on a weighted average update (Izadi et al., 2011; Newcombe et al., 2011):

$$D^{(t+1)} = \frac{W_d^{(t)} D^{(t)} + w_d^{(t+1)} d^{(t+1)}}{W_d^{(t)} + w_d^{(t+1)}} \tag{17}$$

$$W_d^{(t+1)} = \min(W_d^{(t)} + w_d^{(t+1)}, w_{d,max}) \tag{18}$$

Here, D and W denote the fused TSDF value and weight stored in the voxels. The observed truncated signed distance $d^{(t+1)}$ is computed from the depth and its confidence in the fusion process is expressed by its weight $w_d^{(t+1)}$ to obtain high-quality reconstructions.

Instead of storing the observed color value, we store the estimated albedo similar to the approach of Meka et al. (2017). However, we use a different weighting scheme that also accounts for the confidence of the albedo values. With increasing time, the albedo values are refined until they converge to an optimum. We model this process by the confidence function

$$w_{TV}(n) = 1 - \exp\left(-\frac{n}{\sigma_{TV}} \right) \tag{19}$$

measuring the probability that the results have converged after n iterations. Besides the color albedo map, we also store its confidence map and also propagate it from frame to frame. The confidence map can be efficiently updated on-the-fly without explicitly storing the previous number of iterations n. The weight after $c$additional iterations is then given by:

$$w_{TV}(n + c) = 1 + (w_{TV}(n) - 1) \cdot \exp\left(-\frac{c}{\sigma_{TV}} \right) \tag{20}$$

We use these weights to reduce the influence of the albedo values during fusion if they have not yet converged:

$$w_\kappa^{(t+1)} = w_d^{(t+1)} \cdot w_{TV}^{(t+1)} \tag{21}$$

This requires storing a second weight value in each voxel leading to 16 bytes per voxel. The model obtained after fusing depth and albedo information and the extraction of the isosurface may be shown to the user as a direct feedback during the capturing progress.

7. Energy optimization

In this section, we describe the framework used for the optimization of the aforementioned energy functionals. To allow an efficient optimization, we propose an approximate total variation approach (Section 7.1) as well as an additional acceleration strategy for the TV solver that exploits the parallelism offered by GPUs and results in an optimization in real-time (Section 7.2).

7.1. Approximate total variation optimization

Algorithm 1. Primal-Dual-Solver (Chambolle and Pock, 2011)

```
Initialize variables $\boldsymbol{u}^{(0)} \in \mathcal{X}^k$, $\boldsymbol{P}^{(0)} \in \mathcal{Y}_1^k$ and set $\bar{\boldsymbol{u}}^{(0)} = \boldsymbol{u}^{(0)}$
Choose scalars $\sigma, \tau > 0$ and $\theta \in [0, 1]$
for $n = 0, 1, 2, \ldots$ do
    $\boldsymbol{P}^{(n+1)} = prox_{\sigma \delta_{\mathcal{Y}_w^k}}(\boldsymbol{P}^{(n)} + \sigma \nabla \bar{\boldsymbol{u}}^{(n)})$
    $\boldsymbol{u}^{(n+1)} = prox_{\tau E_d}(\boldsymbol{u}^{(n)} - \tau \nabla^{\top} \boldsymbol{P}^{(n+1)})$
    $\bar{\boldsymbol{u}}^{(n+1)} = \boldsymbol{u}^{(n+1)} + \theta(\boldsymbol{u}^{(n+1)} - \boldsymbol{u}^{(n)})$
end for
```

Optimizing the energy functionals in Sections 5.1, 5.2 and 5.3 requires a robust solver that is able to compute optimal solutions in real-time. For this purpose, we use the approach of Chambolle and Pock (2011). We are given an energy function

$$E(\boldsymbol{u}) = E_d(\boldsymbol{u}) + \|w \cdot \boldsymbol{u}\|_1 \tag{22}$$

where E_d represents an arbitrary data term such as a least-squares data fitting term and w denotes a weight map. The image $\boldsymbol{u} \in \mathcal{U}^k$ is an element of the space of images $\mathcal{U}^k$ and stores in each pixel $\boldsymbol{p}$ an image value $\boldsymbol{u}(\boldsymbol{p}) \in \mathbb{R}^k$. This energy $E(\boldsymbol{u})$ can be reformulated in its primal-dual formation

$$E(\boldsymbol{u}, \boldsymbol{P}) = E_d(\boldsymbol{u}) + \langle \nabla \boldsymbol{u} \,|\, \boldsymbol{P} \rangle - \delta_{\mathcal{Y}_w^k}(\boldsymbol{P}) \tag{23}$$

where δ evaluates to zero if $\boldsymbol{P} \in \mathcal{Y}_w^k$ and otherwise to infinity. The image $\boldsymbol{P}$ with values $\boldsymbol{P}(\boldsymbol{p}) \in \mathbb{R}^{k \times 2}$ represents the dual variable to its primal counterpart $\boldsymbol{u}$. The delta function enforces that the dual variable satisfies the constrains of the image space $\mathcal{Y}_w^k$:

$$\forall\, \boldsymbol{p}: \|\boldsymbol{P}(\boldsymbol{p})\|_F \leqslant w(\boldsymbol{p}) \tag{24}$$

Here, $\|\cdot\|_F$ denotes the Frobenius norm. A solution of the primal-dual energy is found by minimizing the primal variable $\boldsymbol{u}$ and maximizing the dual variable $\boldsymbol{P}$ by applying Algorithm 1. After initialization, a gradient ascent update step is performed for $\boldsymbol{P}$, which is followed by a gradient descent step for $\boldsymbol{u}$. Both updates are wrapped with the proximal operator

$$prox_{\lambda f}(\boldsymbol{v}) = \arg\min_{\boldsymbol{u}} \frac{\|\boldsymbol{u} - \boldsymbol{v}\|_2^2}{2\lambda} + f(\boldsymbol{u}) \tag{25}$$

which controls the trade-off between being close to the argument $\boldsymbol{v}$ and minimizing the function f. In case of the dual update step, the proximal operator reduces to a projection $\boldsymbol{\Pi}_{\mathcal{Y}_w^k}$ to the image space $\mathcal{Y}_w^k$ according to

$$\boldsymbol{\Pi}_{\mathcal{Y}_w^k}(\boldsymbol{P}(\boldsymbol{p})) = \frac{\boldsymbol{P}(\boldsymbol{p})}{\max\left(1, \frac{\|\boldsymbol{P}(\boldsymbol{p})\|_F}{w(\boldsymbol{p})}\right)} \tag{26}$$

On the other hand, the optimization in the primal domain is given by

$$\boldsymbol{u}^{(n+1)} = \arg\min_{\boldsymbol{u}} \frac{\|\boldsymbol{u} - (\boldsymbol{u}^{(n)} - \tau \nabla^{\top} \boldsymbol{P}^{(n+1)})\|_2^2}{2\tau} + E_d(\boldsymbol{u}) \tag{27}$$

Since the data term E_d can be arbitrarily complex, the local optimizer depends on the partial derivative of the data term with respect to the new image $\boldsymbol{u}^{(n+1)}$. Therefore, a costly equation system has to be solved for each pixel in general.

To make the approach feasible for arbitrary data terms, we propose to evaluate the data derivative at the current image $\boldsymbol{u}^{(n)}$ instead of the new unknown one. In this case, the primal update step simplifies to

$$\boldsymbol{u}^{(n+1)} = \boldsymbol{u}^{(n)} - \tau\left(\frac{\partial}{\partial \boldsymbol{u}^{(n)}} E_d(\boldsymbol{u}^{(n)}) - div \boldsymbol{P}^{(n+1)}\right) \tag{28}$$

where we apply the well-known identity $\nabla^{\top} = -div$. Algorithm 2 summarizes our novel approximate total variation solver.

Algorithm 2. Our Novel Approximate Primal-Dual-Solver

```
1: Initialize variables $\boldsymbol{u}^{(0)} \in \mathcal{X}^k$, $\boldsymbol{P}^{(0)} \in \mathcal{Y}_1^k$ and set $\bar{\boldsymbol{u}}^{(0)} = \boldsymbol{u}^{(0)}$
2: Choose scalars $\sigma, \tau > 0$ and $\theta \in [0, 1]$
3: for $n = 0, 1, 2, \ldots$ do
4:     $\boldsymbol{P}^{(n+1)} = \boldsymbol{\Pi}_{\mathcal{Y}_w^k}(\boldsymbol{P}^{(n)} + \sigma \nabla \bar{\boldsymbol{u}}^{(n)})$
5:     $\boldsymbol{u}^{(n+1)} = \boldsymbol{u}^{(n)} - \tau\left(\frac{\partial}{\partial \boldsymbol{u}^{(n)}} E_d(\boldsymbol{u}^{(n)}) - div \boldsymbol{P}^{(n+1)}\right)$
6:     $\bar{\boldsymbol{u}}^{(n+1)} = \boldsymbol{u}^{(n+1)} + \theta(\boldsymbol{u}^{(n+1)} - \boldsymbol{u}^{(n)})$
7: end for
```

7.2. *TV solver acceleration*

Our methods proposed for albedo estimation and image segmentation are all built upon a state-of-the-art total variation solver (Chambolle and Pock, 2011) that can efficiently be run on the GPU. However, several hundreds of iterations are typically needed to reach convergence which prevents the optimization from being feasible in real time. To overcome this issue, we exploit the high frame rate of current commodity sensors such as the Kinect v2. Consecutive frames only differ slightly in the viewing angle so that most parts of an image are also visible in the next ones. Many state-of-the-art 3D reconstruction algorithms use this overlap to estimate the camera pose with respect to a global coordinate system (Izadi et al., 2011; Newcombe et al., 2011). Given the poses $\boldsymbol{T}^{(t)}$, $\boldsymbol{T}^{(t+1)}$ inferred from our underlying reconstruction pipeline and the new depth map $z^{(t+1)}$ allows to forward project an image $I^{(t)}$ from the current time step t to the new time step $t+1$:

$$\widetilde{I}^{(t+1)}(\boldsymbol{p}_i) = I^{(t)}(\boldsymbol{p}_{c(i)}) \tag{29}$$

The corresponding pixel $\boldsymbol{p}_{c(i)}$ is obtained by transforming each vertex $\boldsymbol{v}(z^{(t+1)}, \boldsymbol{p}_i)$ from the local camera coordinate system at time step $t+1$ to the one at time step t and perspectively projecting it onto the image plane. This operation can be computed in constant time per pixel.

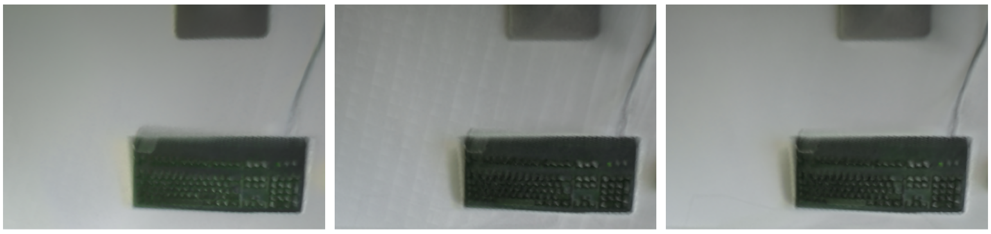
We use the forward projection technique to propagate the solutions of each TV solver to the next frame. These solutions hence serve as an initial guess and are used to reinitialize the solvers so that the results are continuously refined over time. The additional degree of the freedom gained here by distributing the computational load across subsequent frames relaxes the hardware requirements and allows real-time performance of our approach even with non-high-end GPUs.

8. Results

Implementation details. All experiments were performed on an Intel Core i7-4930K with 32 GB RAM and a Nvidia GeForce GTX TITAN X with 12 GB VRAM. In all experiments, the following parameter values were used for the albedo estimation: $\lambda_{d,IR} = 3.0$, $\sigma_{d,IR} = 0.25$, $\lambda_{data,RGB} = 3.0$, $\lambda_{coup,RGB} = 3.0$, $\lambda_{reg,RGB} = 1.0$, $\sigma_{d,RGB} = 0.01$, $\lambda_{seg} = 2.0$, $\sigma_{seg} = 0.25$, $\sigma_{TV} = 150.0$. For the total variation solver, we always set the primal and dual step width to $\sigma = \tau = 0.025$ and the extrapolation weight to $\theta = 1.0$. All these values were determined heuristically and provided stable results for all of the considered scenarios. Furthermore, the 3D space was discretized with a voxel resolution of 5 mm as applied by several related geometry reconstruction techniques (e.g. Izadi et al. (2011), Newcombe et al. (2011), Nießner et al. (2013)). The higher memory requirements of our approach in comparison to previous approaches (Nießner et al., 2013; Meka et al., 2017) resulting from the need for storing the estimated albedo values and a corresponding fusion weight, however, do not represent a significant limitation as most real-world scenes easily fit in the available GPU memory due to the sparse nature of the geometry-related part of our approach. On average, the geometry-related part of the pipeline took 14.9 ms where the camera pose estimation took 7.5 ms, the fusion of geometry and albedo information 3.9 ms, and the final isosurface extraction 3.5 ms.

Datasets. We tested the performance of our approach on several indoor scenes captured with a Kinect v2. Since most datasets on intrinsic image decomposition do not provide IR images, a fair comparison to other approaches that do not take advantage of such data is hardly achievable. In the following, we will compare our work against the approach by Kerl et al. (2014), that also exploits the IR data provided by the Kinect sensor, and demonstrate how the results can be improved based on our approach. However, taking advantage of additional IR data makes us believe that a comparison against approaches that only rely on RGB or RGB-D data would not be fair.

TV solver scalability. We compared the performance of our dedicated total variation solver with conventional approaches. For this purpose, we evaluated the reflectance-related part of our approach, which includes the image segmentation and the estimation of the albedo image in the IR and RGB image, without propagating intermediate results between frames. 300 iterations for each of the three optimization tasks were used leading to a total computation time of 889.7 ms per frame on



(a) 300 iterations

(b) 6 iterations with forward projection

(c) 6 iterations with forward projection and confidence-based weighting

Fig. 4. Comparison of reconstructed models. Simple forward projection already allows the use of significantly less iterations and improves the run-times dramatically (889.7 ms vs. 21.3 ms) but leads to a poor reconstruction quality (b). Our confidence based weighting combined with forward projection (c) overcomes this limitation without increasing the number of iterations and generates high-quality reconstructions that are otherwise only possible with a significantly higher number of iterations (a). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

average. Compared to typical run-times of CPU versions (Chambolle and Pock, 2011), this is a substantial improvement by a factor of about 40, but still far from real-time rates.

Our image propagation approach through forward projection dramatically reduces the computational costs and requires a significantly smaller number of iterations per frame for convergence. In our experiments, we only used 6 iterations for each step leading to a total estimation time of 21.3 ms on average which is well within real-time rates. However without appropriately handling non-converged parts of the albedo images, several artifacts may be introduced into the final reconstruction leading to a poor reconstruction quality as shown in Fig. 4. Our full pipeline including forward projection and confidence-based weighting did not suffer from such artifacts and yields similar results as obtained with a significantly higher number of iterations. This demonstrates that the additional degree of freedom gained through propagation can be used to adjust the number of iterations depending on the power of the GPU to achieve real-time performance even on non-high-end hardware.

Intrinsic video decomposition. Since our albedo estimation approach without fusion can also be used for intrinsic video decomposition, we can apply it to several other applications such as recoloring, relighting, material editing, etc. (Meka et al., 2016; Bonneel et al., 2017). In a supplemental video, we show a relighting application where a movable light source is added into the scene and the modified object appearance is rendered in real-time. While we could leverage the high-resolution RGB image data of the Kinect v2 in such scenarios, we instead applied it to RGB-D and IR data given in the depth camera's resolution to directly show the results used for albedo fusion. Furthermore, we also demonstrate the scalability of our dedicated total variation solver in the supplemental video which makes our approach run in real-time similar to the L2 regularization approach of Meka et al. (2016, 2017), however, at the increased robustness provided by an L1 regularization.

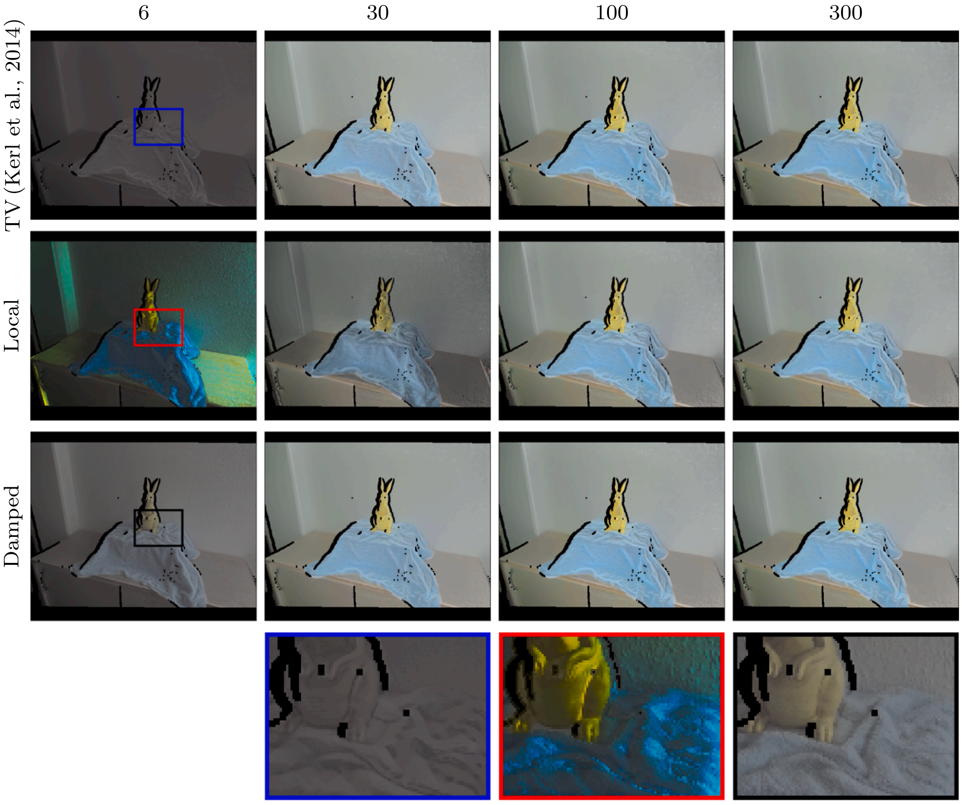


Fig. 5. Comparison of albedo maps obtained with different energy priors and different numbers of iterations. Whereas previous approaches (Kerl et al., 2014) alternately optimize both the albedo and the shading image using Total Variation (first row), our damped local optimizer (third row) provides the improved run-time of a local optimizer (second row) and at the same time a higher convergence rate when using fewer iterations. The corresponding shading images and run-time measurements are shown in Fig. 6 and Table 2 respectively. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

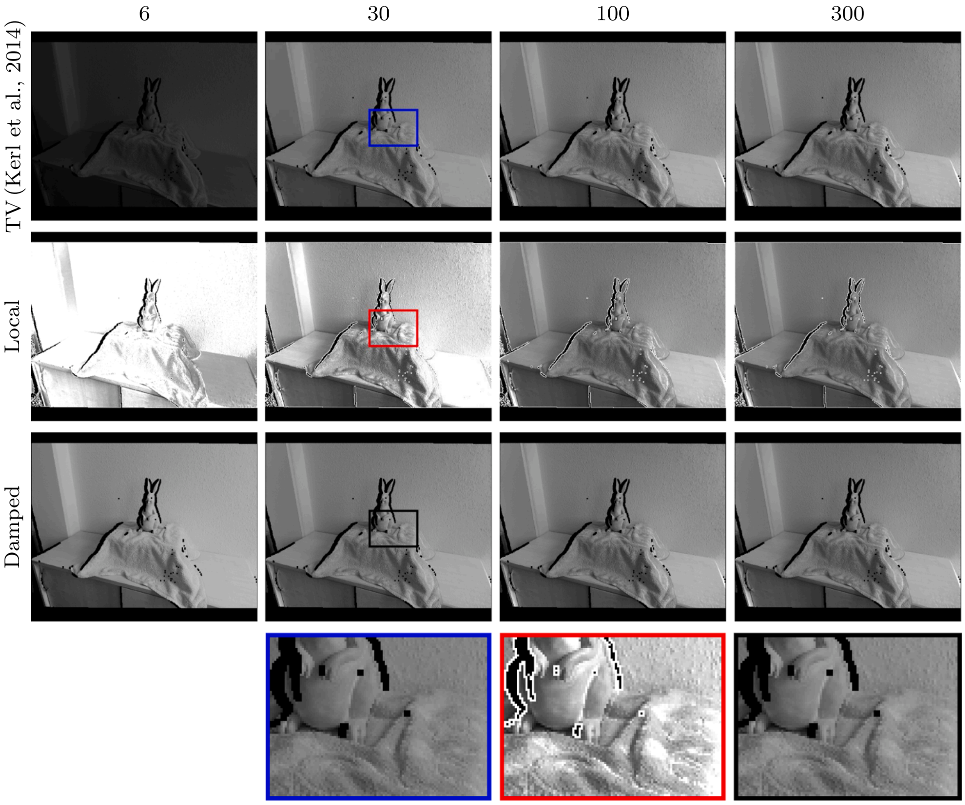


Fig. 6. Comparison of shading maps obtained with different energy priors and number of iterations corresponding to the color albedo maps shown in Fig. 5. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Shading map optimization. In addition to the improved total variation solver framework, we also compared our novel energy formulation regarding the color shading map. To evaluate the convergence speed and the run-time, we only considered the RGB and IR reflectance reconstruction steps and discarded the image segmentation and forward projection steps. Therefore, the number of iterations needed to estimate the infrared albedo image with the total variation solver was increased from 300 to provide converged estimates to its color counterpart. Both the albedo and shading map are initialized with zeros to simulate missing data during forward projection and highlight how effective these strategies perform in such cases. The results are shown in Figs. 5 and 6 and Table 2.

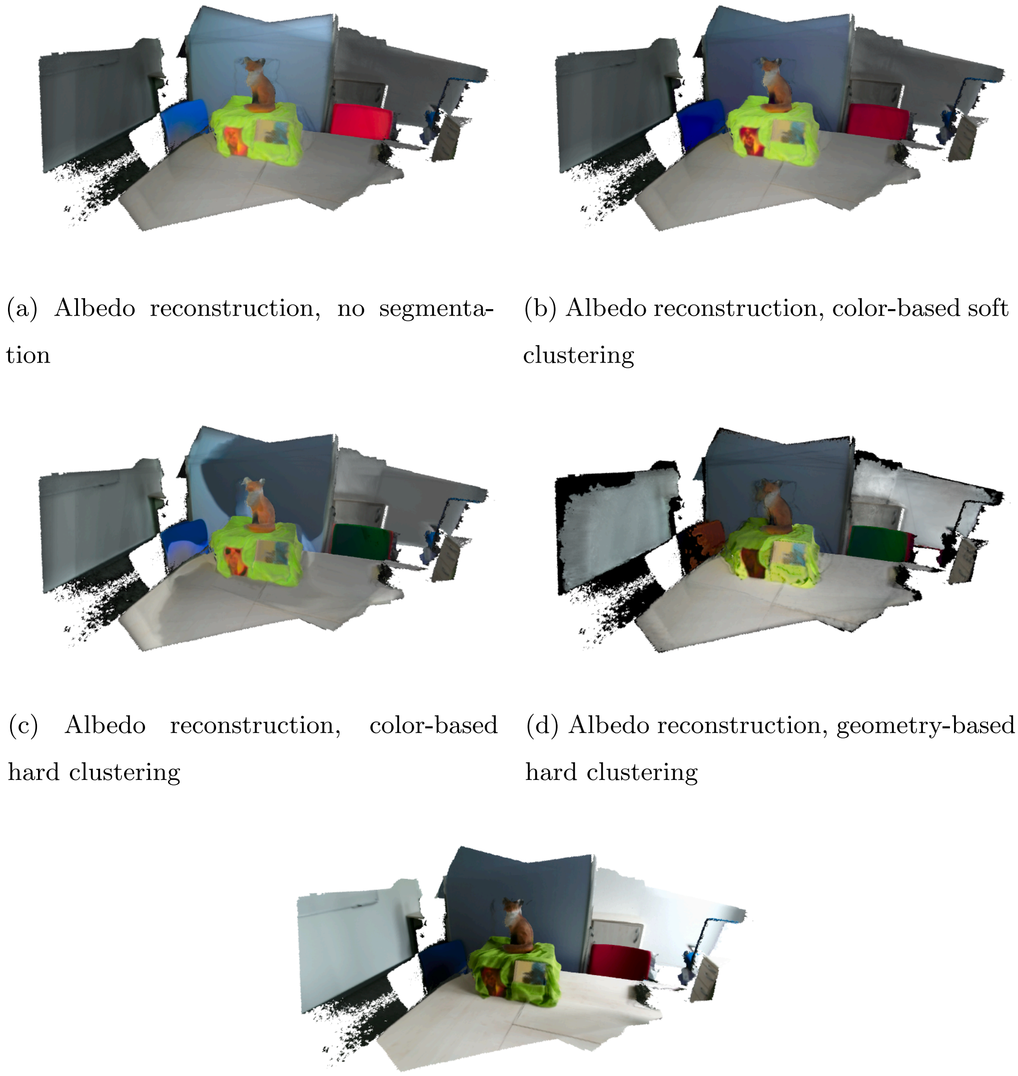
Applying a Total Variation optimization for both the color albedo and the color shading map leads to a slow convergence rate at fewer iterations and a moderate rate at a higher number of iterations. A pure local solver leads to similar results at convergence and improved run-times by up to 20%. However, intermediate results suffer from strong artifacts such as wrong albedo estimates at few iterations (see entry (Local, 6) in Fig. 5) and implausible values at object boundaries (see middle row in Fig. 6). Since only the data term affects the shading map, the corresponding error term does not contribute to the albedo optimization and errors due to missing data at object boundaries cannot be reduced or eliminated. Our damped energy functional (see bottom row of Fig. 6) avoids such artifacts and provides faster run-times and better convergence rates in comparison to the other approaches.

Table 2
Run-time measurements for different energy priors. We compared the run-time for different energy regularization priors during the shading image optimization with respect to the number of iterations. While the total variation prior significantly consumes more run-time, our damping prior has no impact on the performance and provides similar or better convergence results as shown in Figs. 5 and 6.

Energy	Run-time [ms]			
	6	30	100	300
TV	7.9	30.4	92.5	271.8
Local	7.3	**25.7**	78.8	**229.9**
Damped	**7.2**	25.9	**78.6**	230.0

Segmentation. We also tested different segmentation strategies to provide a more robust coupling between infrared and color albedo images. For the color-based hard and soft clustering approaches, we used a predefined number of five segments and initialized four of them along the two axes of the *a* and *b* components of the LAB color space and the last one in the center which reduces the probability that two clusters immediately fall together. Since we use segmentation to improve the coupling, we observed that five segments are sufficient to handle the different ratios between mean infrared albedo and mean color radiance and that a full object segmentation is not required. The geometry-based technique automatically adjusts the number of clusters during reconstruction, so we only cap the maximum number to 100. The corresponding results for the final reconstructions of geometry and albedo as well as for one image inside the captured image sequence are shown in Figs. 7 and 8 respectively. The segmentations of the selected image are shown in Fig. 9.

Since the coupling factors in the albedo estimation framework largely affect the quality of the results, artifacts caused by incorrect segmentations are directly propagated to the color albedo maps. Color-based hard clustering might fail to fully segment objects from each other leading to under-segmentation. On the other hand, geometry-based approaches (e.g. Tateno et al. (2016)) tend to over-segment the scene. Furthermore, scene parts that are segmented as object boundaries are considered to be holes in the albedo estimation step. Segmentation failures might also lead to inconsistently fused albedo textures as shown in Fig. 7c and d. Our soft segmentation approach (see



(a) Albedo reconstruction, no segmentation

(b) Albedo reconstruction, color-based soft clustering

(c) Albedo reconstruction, color-based hard clustering

(d) Albedo reconstruction, geometry-based hard clustering

(e) Color texture reconstruction

Fig. 7. Virtual 3D models reconstructed from RGB-D and IR data. Artifacts and incorrect results obtained from hard clustering approaches directly propagate to the color albedo reconstruction. On the other hand, soft clustering avoids such artifacts and leads to more expressive models than previous approaches that fuse colors directly. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Fig. 7b) does not suffer from such artifacts and also improves the quality especially for objects being shadowed where the observed RGB values are close to black (see close-ups of Fig. 8). The corresponding albedo reconstruction shows the highest quality even preserving the structures and color gradients within the scene as it can e.g. be observed on the book covers surrounded by the green cloth (see Fig. 7).

9. Discussion

We demonstrated that our novel framework is capable of automatically reconstructing high-quality surface geometry together with surface reflectance information in terms of albedo information at real-time rates. As shown in the previous section, our improved total variation solver exploits the high frame rate of the Kinect sensor to improve run-time performance by up to a factor of 40. Our novel energy formulation based on soft scene segmentation and optimized regularization priors not only leads to better decomposition results (see Fig. 8) but also converges faster in a significantly smaller number of iterations in comparison to previous work (Kerl et al., 2014).

While we have also demonstrated that our approach is more robust in comparison to previous approaches in terms of a more flexible separation into albedo and shading layers, it still has a few limitations. The direct coupling of IR and RGB albedo values sometimes leads to undesirable results as estimation artifacts and imperfections in the infrared albedo image are propagated into the color albedo map. This may be addressed by a more expressive infrared reflectance model that better fits to the observed radiance values. Furthermore, our segment-based coupling technique assumes that the estimated segment borders correspond to edges in the albedo image where the surface reflectance properties change. In scenes with complex illumination conditions and self-shadowing, objects with a uniform albedo might be accidentally segmented into multiple or incorrect segments (see right column of Fig. 11) leading to possibly different albedo estimates. Color textures that are invisible in the infrared channel violate the assumption that the appearance in both the RGB and IR channel is similar and result in a flat reconstructed reflectance that matches the one of the underlying material. Although our soft segmentation technique also improves the decomposition robustness for such objects (see top left row of Fig. 11), such scenes are still challenging for IR-based approaches. For most real-world materials such as wood, cloth, plastic, etc., however, this

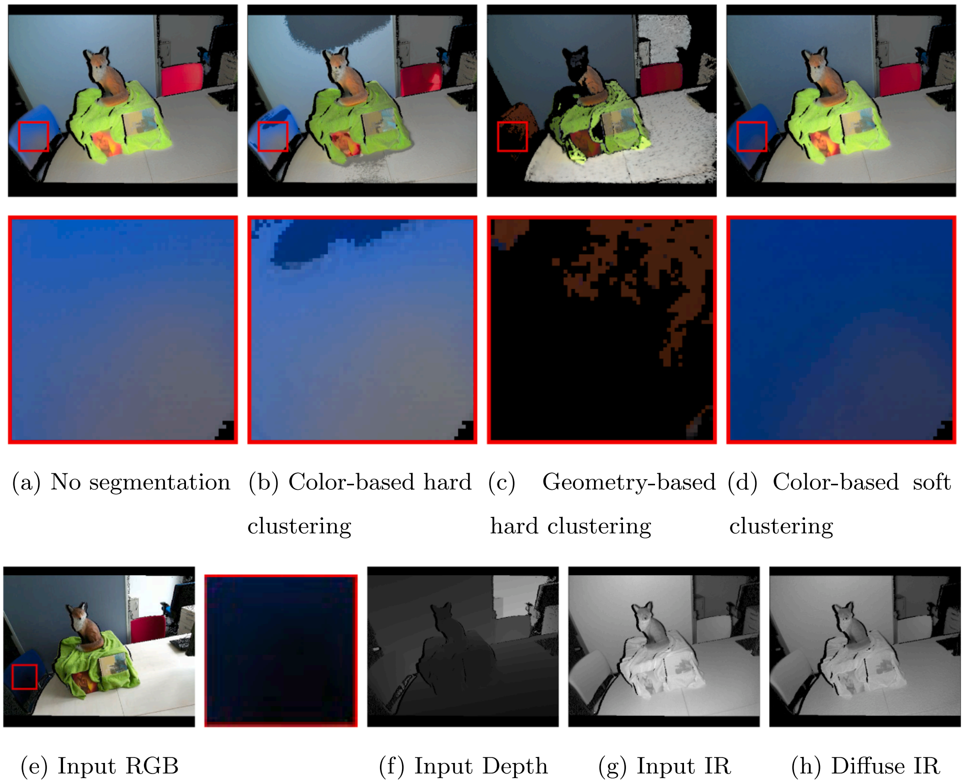

(a) No segmentation
(b) Color-based hard clustering
(c) Geometry-based hard clustering
(d) Color-based soft clustering
(e) Input RGB
(f) Input Depth
(g) Input IR
(h) Diffuse IR

Fig. 8. Comparison between segmentation approaches. Whereas hard clustering techniques ((b) and (c)) may introduce artifacts due to wrong segmentation or coupling, soft clustering achieves smooth and more accurate results (d) in comparison to image-wise coupling (a). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

assumption holds and, hence, leads to improved decomposition results. For specular scenes, undesired shading effects such as highlights in the IR domain might be propagated to the estimated diffuse RGB albedo image (see bottom left row of Fig. 11). Furthermore, small errors in the camera poses due to a non-optimal registration of subsequent frames may result in high-frequency shading edges being not fully factored into the shading image (see Fig. 10). However, such complex illumination can also not be handled correctly by previous work relying on low-frequency spherical harmonics or similar representations. Finally, our acceleration technique might introduce some artifacts in the final reconstruction that were not fully compensated by our novel weighting scheme. However, our total variation solver is several orders of magnitude faster than previous approaches which may allow enough capacity for further processing steps. For this reason we believe our approach to be of great relevance for future developments in this context.

10. Conclusions

We presented a novel approach to simultaneously reconstruct geometry and surface albedo properties of a scene with a Kinect v2. The captured RGB-D and IR data are used to efficiently compute the reflectance in the infrared channel and to segment the image space into a set of clusters based on soft clustering. Our novel segment-wise formulation of the intrinsic image decomposition problem incorporates these information to improve the robustness of the obtained decomposition results for albedo and shading components and, finally, the inferred albedo information is fused into the reconstructed 3D model. Using a dedicated total variation solver that exploits the high framerate of the Kinect, we demonstrated in various indoor scenes that our framework is able to generate high-quality real-time reconstructions in a fully automatic manner. Therefore, we believe that our work

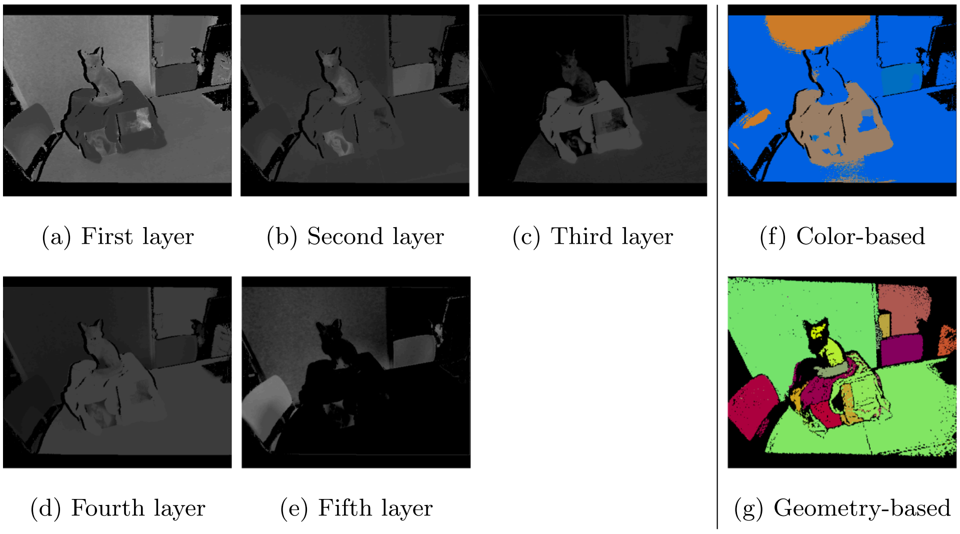


Fig. 9. Soft and hard clustering results. Each image of the five images (left) decodes the probability of a pixel to be associated with its corresponding coupling factor. For the other two images, pixel assignments to hard clustered segments are color coded (right). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

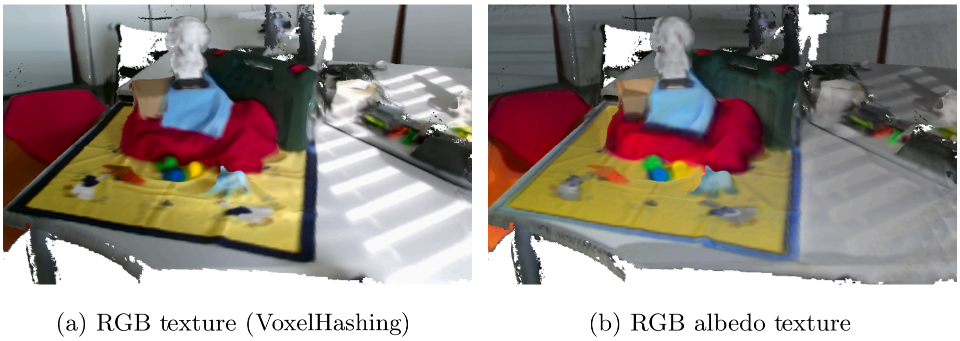

(a) RGB texture (VoxelHashing)
(b) RGB albedo texture

Fig. 10. In contrast to only reconstructing a color texture in conjunction with a virtual 3D model, which may contain illumination artifacts (a), our technique reconstructs point-wise color albedo values that are smooth even in challenging scenarios with complex illumination (b). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

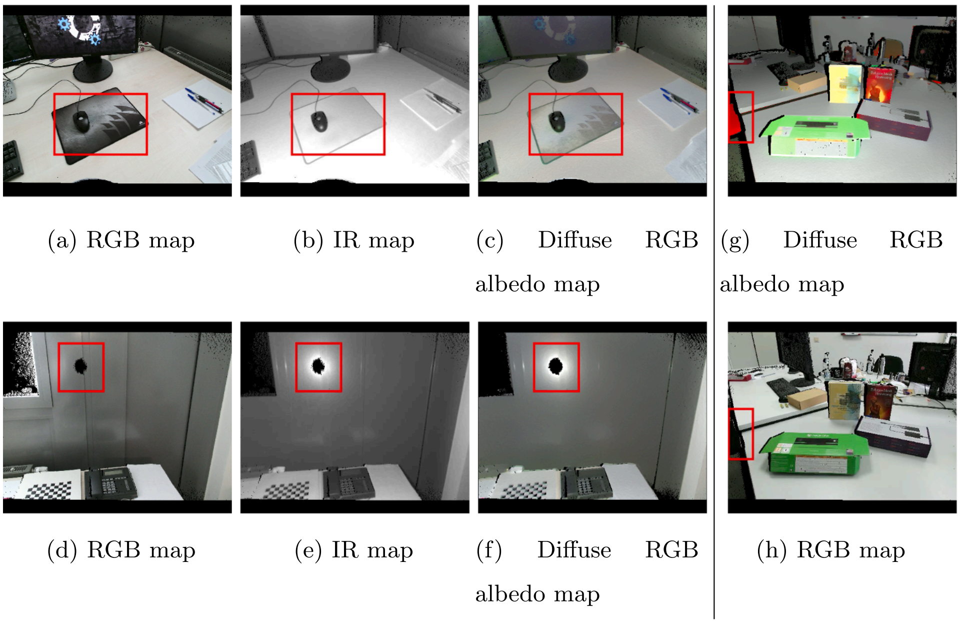

(a) RGB map
(b) IR map
(c) Diffuse RGB albedo map
(g) Diffuse RGB albedo map
(d) RGB map
(e) IR map
(f) Diffuse RGB albedo map
(h) RGB map

Fig. 11. Limitations of our technique. Unexpected (invisible textures, top left row) as well as undesired (highlights, bottom left row) effects or assignments to incorrect segments (right column) may lead to less optimal results during albedo estimation. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

represents a significant step towards capturing realistic environments that enable a better immersive experience of objects and are required for new augmented and virtual reality applications.

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).

Appendix A. Supplementary material

Supplementary data associated with this article can be found, in the online version, at https://doi.org/10.1016/j.isprsjprs.2019.01.018.

References

Barron, J.T., Malik, J., 2013. Intrinsic Scene Properties from a Single RGB-D Image. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, pp. 17–24.
Barron, J.T., Malik, J., 2015. Shape, illumination, and reflectance from shading. IEEE Trans. Pattern Anal. Mach. Intell. 37, 1670–1687.
Bi, S., Kalantari, N.K., Ramamoorthi, R., 2017. Patch-based optimization for image-based texture mapping. ACM Trans. Graph. (Proc. SIGGRAPH 2017 36, 106:1–106:11.
Bonneel, N., Kovacs, B., Paris, S., Bala, K., 2017. Intrinsic decompositions for image editing. Comput. Graph. Forum 36, 593–609.
Chambolle, A., Pock, T., 2011. A first-order primal-dual algorithm for convex problems with applications to imaging. J. Math. Imag. Vision 40, 120–145.
Chen, J., Bautembach, D., Izadi, S., 2013. Scalable real-time volumetric surface reconstruction. ACM Trans. Graph. 32, 113:1–113:16.
Choe, G., Park, J., Tai, Y.W., Kweon, I.S., 2017. Refining geometry from depth sensors using IR shading images. Int. J. Comput. Vision 122, 1–16.
Choe, G., Park, J., Tai, Y.W., So Kweon, I., 2014. Exploiting Shading Cues in Kinect IR Images for Geometry Refinement. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3922–3929.
Curless, B., Levoy, M., 1996. A volumetric method for building complex models from range images. In: Proceedings of the Annual Conference on Computer Graphics and Interactive Techniques, pp. 303–312.
Dai, A., Nießner, M., Zollhöfer, M., Izadi, S., Theobalt, C., 2017. BundleFusion: real-time globally consistent 3D reconstruction using on-the-fly surface reintegration. ACM Trans. Graph. (TOG) 36, 24:1–24:18.
Fankhauser, P., Bloesch, M., Rodriguez, D., Kaestner, R., Hutter, M., Siegwart, R.Y., 2015. Kinect v2 for mobile robot navigation: evaluation and modeling. In: International Conference on Advanced Robotics (ICAR). IEEE, pp. 388–394.
Guarnera, D., Guarnera, G.C., Ghosh, A., Denk, C., Glencross, M., 2016. BRDF Representation and Acquisition. In: Computer Graphics Forum. Wiley Online Library, pp. 625–650.
Guo, K., Xu, F., Yu, T., Liu, X., Dai, Q., Liu, Y., 2017. Real-time geometry, albedo, and motion reconstruction using a single RGB-D Camera. ACM Trans. Graph. (TOG) 36.
Hachama, M., Ghanem, B., Wonka, P., 2015. Intrinsic Scene Decomposition from RGB-D Images. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV). IEEE Computer Society, pp. 810–818.
Izadi, S., Kim, D., Hilliges, O., Molyneaux, D., Newcombe, R., Kohli, P., Shotton, J., Hodges, S., Freeman, D., Davison, A., Fitzgibbon, A., 2011. KinectFusion: Real-time 3D Reconstruction and Interaction Using a Moving Depth Camera. In: Proceedings of the Annual ACM Symposium on User Interface Software and Technology. ACM, pp. 559–568.
Jin, X., Gu, Y., 2017. Superpixel-based intrinsic image decomposition of hyperspectral images. IEEE Trans. Geosci. Remote Sens. 55, 4285–4295.
Kähler, O., Prisacariu, V., Valentin, J., Murray, D., 2016a. Hierarchical voxel block hashing for efficient integration of depth images. IEEE Robot. Autom. Lett. 1, 192–197.
Kähler, O., Prisacariu, V.A., Murray, D.W., 2016b. Real-Time Large-Scale Dense 3D Reconstruction with Loop Closure. In: European Conference on Computer Vision. Springer, pp. 500–516.
Kähler, O., Prisacariu, V.A., Ren, C.Y., Sun, X., Torr, P., Murray, D., 2015. Very high frame rate volumetric integration of depth images on mobile devices. IEEE Trans. Visual.

Comput. Graph. 21, 1241–1250.

Kajiya, J.T., 1986. The rendering equation. In: ACM Siggraph Computer Graphics. ACM, pp. 143–150.

Kerl, C., Souiai, M., Sturm, J., Cremers, D., 2014. Towards Illumination-invariant 3D Reconstruction using ToF RGB-D Cameras. In: International Conference on 3D Vision (3DV). IEEE, pp. 39–46.

Lambert, J.H., 1760. Photometria sive de mensura et gradibus luminis, colorum et umbrae. Klett.

Li, S., Handa, A., Zhang, Y., Calway, A., 2016. HDRFusion: HDR SLAM using a low-cost auto-exposure RGB-D sensor. In: International Conference on 3D Vision (3DV). IEEE, pp. 314–322.

Liu, Y., Gao, W., Hu, Z., 2018. Geometrically stable tracking for depth images based 3D reconstruction on mobile devices. ISPRS J. Photogram. Remote Sens. 143, 222–232.

Maier, R., Kim, K., Cremers, D., Kautz, J., Nießner, M., 2017. Intrinsic3D: High-Quality 3D Reconstruction by Joint Appearance and Geometry Optimization with Spatially-Varying Lighting. In: Proceedings of the IEEE International Conference on Computer Vision. IEEE Computer Society, pp. 3133–3141.

Meka, A., Fox, G., Zollhöfer, M., Richardt, C., Theobalt, C., 2017. Live user-guided intrinsic video for static scene. IEEE Trans. Visual. Comput. Graph. 23, 2447–2454.

Meka, A., Maximov, M., Zollhoefer, M., Chatterjee, A., Seidel, H.P., Richardt, C., Theobalt, C., 2018. LIME: Live Intrinsic Material Estimation. In: Proceedings of Computer Vision and Pattern Recognition (CVPR), pp. 6315–6324.. http://gvv.mpi-inf.mpg.de/projects/LIME/ .

Meka, A., Zollhöfer, M., Richardt, C., Theobalt, C., 2016. Live intrinsic video. ACM Trans. Graph. (TOG) 35, 109:1–109:14.

Newcombe, R.A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A.J., Kohi, P., Shotton, J., Hodges, S., Fitzgibbon, A., 2011. KinectFusion: Real-Time Dense Surface Mapping and Tracking. In: IEEE International Symposium on Mixed and augmented reality (ISMAR). IEEE, pp. 127–136.

Nießner, M., Zollhöfer, M., Izadi, S., Stamminger, M., 2013. Real-time 3D reconstruction at scale using voxel hashing. ACM Trans. Graph. (TOG) 32, 169:1–169:11.

Or-El, R., Hershkovitz, R., Wetzler, A., Rosman, G., Bruckstein, A.M., Kimmel, R., 2016. Real-time Depth Refinement for Specular Objects. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4378–4386.

Pagliari, D., Pinto, L., 2015. Calibration of Kinect for Xbox one and comparison between the two generations of Microsoft sensors. Sensors 15, 27569–27589.

Payne, A., Daniel, A., Mehta, A., Thompson, B., Bamji, C.S., Snow, D., Oshima, H., Prather, L., Fenton, M., Kordus, L., O'Connor, P., McCauley, R., Nayak, S., Acharya, S., Mehta, S., Elkhatib, T., Meyer, T., O'Dwyer, T., Perry, T., Xu, Z., 2014. A 512 × 424 CMOS 3D Time-of-Flight Image Sensor with Multi-Frequency Photo-Demodulation up to 130MHz and 2GS/s ADC. In: IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC), pp. 134–135.

Rajput, A., Funk, E., Börner, A., Hellwich, O., 2018. A regularized volumetric fusion framework for large-scale 3D reconstruction. ISPRS J. Photogram. Remote Sens. 141, 124–136.

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

Shi, J., Dong, Y., Tong, X., Chen, Y., 2015. Efficient Intrinsic Image Decomposition for RGBD Images. In: Proceedings of the ACM Symposium on Virtual Reality Software and Technology. ACM, pp. 17–25.

Stotko, P., 2016. State of the Art in Real-time Registration of RGB-D Images. In: Proceedings of the Central European Seminar on Computer Graphics, pp. 155–170.

Tateno, K., Tombari, F., Navab, N., 2016. When 2.5D is not enough: Simultaneous Reconstruction, Segmentation and Recognition on dense SLAM. In: IEEE International Conference on Robotics and Automation (ICRA). IEEE, pp. 2295–2302.

Valgma, L., 2016. 3D reconstruction using Kinect v2 camera. Ph.D. thesis. Tartu Ülikool.

Whelan, T., Kaess, M., Fallon, M., Johannsson, H., Leonard, J., McDonald, J., 2012. Kintinuous: Spatially Extended KinectFusion. In: RSS Workshop on RGB-D: Advanced Reasoning with Depth Cameras, Sydney, Australia.

Whelan, T., Kaess, M., Johannsson, H., Fallon, M., Leonard, J.J., McDonald, J., 2015. Real-time large-scale dense RGB-D SLAM with volumetric fusion. Int. J. Robot. Res. 34, 598–626.

Whelan, T., Salas-Moreno, R.F., Glocker, B., Davison, A.J., Leutenegger, S., 2016. ElasticFusion: real-time dense SLAM and light source estimation. Int. J. Robot. Res. 35, 1697–1716.

Wu, C., Zollhöfer, M., Nießner, M., Stamminger, M., Izadi, S., Theobalt, C., 2014. Real-time shading-based refinement for consumer depth cameras. ACM Trans. Graph. (TOG) 33, 200.

Wu, H., Wang, Z., Zhou, K., 2016. Simultaneous localization and appearance estimation with a consumer RGB-D Camera. IEEE Trans. Visual. Comput. Graph. 22, 2012–2023.

Wu, H., Zhou, K., 2015. AppFusion: Interactive Appearance Acquisition Using a Kinect Sensor. In: Computer Graphics Forum. Wiley Online Library, pp. 289–298.

Zennaro, S., Munaro, M., Milani, S., Zanuttigh, P., Bernardi, A., Ghidoni, S., Menegatti, E., 2015. Performance evaluation of the 1st and 2nd generation Kinect for multimedia applications. In: IEEE International Conference on Multimedia and Expo (ICME). IEEE, pp. 1–6.

Zhou, Q.Y., Koltun, V., 2014. Color map optimization for 3D reconstruction with consumer depth cameras. ACM Trans. Graph. (TOG) 33, 155:1–155:10.

Zollhöfer, M., Stotko, P., Görlitz, A., Theobalt, C., Nießner, M., Klein, R., Kolb, A., 2018. State of the Art on 3D Reconstruction with RGB-D Cameras. Comput. Graph. Forum 37, 625–652.

Zuo, X., Wang, S., Zheng, J., Yang, R., 2017. Detailed surface geometry and albedo recovery from RGB-D video under natural illumination. IEEE Int. Conf. Comput. Vision (ICCV) 3152–3161.