TOWARDS QUANTITATIVE ENDOSCOPY WITH VISION INTELLIGENCE

by
Xingtong Liu

A dissertation submitted to Johns Hopkins University in conformity with the requirements for the degree of Doctor of Philosophy.

Baltimore, Maryland
September 2021

Abstract

In this thesis, we work on topics related to quantitative endoscopy with vision-based intelligence. Specifically, our works revolve around the topic of video reconstruction in endoscopy, where many challenges exist, such as texture scarceness, illumination variation, multimodality, *etc*., and these prevent prior works from working effectively and robustly. To this end, we propose to combine the strength of expressivity of deep learning approaches and the rigorousness and accuracy of non-linear optimization algorithms to develop a series of methods to confront such challenges towards quantitative endoscopy. We first propose a retrospective sparse reconstruction method that can estimate a high-accuracy and density point cloud and high-completeness camera trajectory from a monocular endoscopic video with state-of-the-art performance. To enable this, replacing the role of a hand-crafted local descriptor, a deep image feature descriptor is developed to boost the feature matching performance in a typical sparse reconstruction algorithm. A retrospective surface reconstruction pipeline is then proposed to estimate a textured surface model from a monocular endoscopic video, where self-supervised depth and descriptor learning and surface fusion technique is involved. We show that the proposed method performs superior to a popular dense reconstruction method and the estimate reconstructions are in good agree-

ment with the surface models obtained from CT scans. To align video-reconstructed surface models with pre-operative imaging such as CT, we introduce a global point cloud registration algorithm that is robust to resolution mismatch that often happens in such multi-modal scenarios. Specifically, a geometric feature descriptor is developed where a novel network normalization technique is used to help a 3D network produce more consistent and distinctive geometric features for samples with different resolutions. The proposed geometric descriptor achieves state-of-the-art performance, based on our evaluation. Last but not least, a real-time SLAM system that estimates a surface geometry and camera trajectory from a monocular endoscopic video is developed, where deep representations for geometry and appearance and non-linear factor graph optimization are used. We show that the proposed SLAM system performs favorably compared with a state-of-the-art feature-based SLAM system.

Primary Reader and Advisor: Mathias Unberath and Russell H. Taylor

Secondary Reader: Gregory D. Hager and Masaru Ishii

Acknowledgments

As I am getting close to the end of the journey towards my Ph.D. degree, my heart is filled with joy, gratitude, fulfillment, and excitement. Along this journey, I received plenty of help and support, which I believe is necessary to make me reach where I am now, from many people to whom I am very grateful. I want to first express my sincere gratitude to my thesis advisors, Mathias Unberath and Russell H. Taylor. When I first began my Ph.D. study, I did not know much about everything required to obtain my Ph.D. degree, especially the ability to do proper research. My advisors provided me a lot of precious and helpful academic and professional advice in terms of both the general directions and the details. Thanks to them, I believe my ability to conduct high-level research projects and write academic papers with high quality has been largely improved throughout the years and my career vision has become clearer. I feel very lucky to have these two excellent people as my thesis advisors. Second, I want to show my gratitude to my wife. We first met at a welcome party for students in Beijing who will go to the Johns Hopkins University to study. Since then, we have spent a lot of joyful time together and became best friends. In January 2019, I proposed to her with a wedding ring and we got married afterward. Whenever I felt exhausted and stressed doing research, writing a paper, chasing a conference

deadline, and waiting for results, she was always by my side and enlightened me and cheered me up with her optimism. With her accompany, I always feel like everything will be alright as long as we are together. She will always be the apple of my eye till death do us part. I was born in a lovely family in Shandong Province, China. Both of my parents are teachers. My mother, Ping Gong, teaches English in high school and my father, Ping Liu, teaches safety instructions of oil and gas well drilling. Thanks to the love from my parents, I had a happy and relaxing childhood life. Their attitude towards education also enlightens me to always try to do my best at study and never surrender to the challenges ahead. I still remember that, when I was a kid, my father would catch grasshoppers for me when he gets back from work and I was always extremely excited about that. I still remember that, when I was in high school, my mother would always get up hours earlier and cook delicious breakfast for me and I can always feel the rich love inside the meal. They are the people who let me come to this beautiful world and they will always be the ones who I love the most. Lastly, I would also like to thank all my committee members, my former advisor, course professors, researchers, colleagues, and friends who have provided valuable help and support during my Ph.D. study: Mathias Unberath, Russell H. Taylor, Gregory D. Hager, Masaru Ishii, Austin Reiter, Jerry Prince, Alan Yuille, Simon Leonard, Chien-Ming Huang, Randal Burns, Xin Jin, Anton Deguet, Cong Gao, Zhaoshuo Li, Ayushi Sinha, Benjamin Killeen, Javad Fotouhi, Yiping Zheng, Maia Stiber, Jindan Huang, and Jieying Wu.

Dedication

This thesis is dedicated to my family, my wife, and the people I love.

Contents

List of Tables

List of Figures

List of Abbreviations

2D	Two **D**imensional
3D	Three **D**imensional
ARD	**A**bsolute **R**elative **D**ifference
ATE	**A**bsolute **T**rajectory **E**rror
BA	**B**undle **A**djustment
BCE	**B**inary **C**ross **E**ntropy
BIN	**B**atch-**I**nstance **N**ormalization
BN	**B**atch **N**ormalization
CCD	**C**harged-**C**oupled **D**evice
CDF	**C**umulative **D**ensity **F**unction
CNN	**C**onvolutional **N**eural **N**etwork
CT	**C**omputed **T**omography
DC	**D**epth **C**onsistency
EES	**E**ndoscopic **E**ndonasal **S**urgery

LIST OF ABBREVIATIONS

FMR	**F**eature-**M**atch **R**ecall
GAN	**G**enerative **A**dversarial **N**etwork
GMM	**G**aussian **M**ixture **M**odel
GN	**G**roup **N**ormalization
GPU	**G**raphics **P**rocessing **U**nit
HC	**H**ardest **C**ontrastive
HSV	**H**ue **S**aturation **V**alue
ICP	**I**terative **C**losest **P**oint
IMLOP	**I**terative **M**ost **L**ikely **O**riented **P**oint
IN	**I**nstance **N**ormalization
IRB	**I**nstitutional **R**eview **B**oards
LCN	**L**ocal **C**ontext **N**ormalization
LM	**L**evenberg-**M**arquardt
LN	**L**ayer **N**ormalization
LRN	**L**ocal **R**esponse **N**ormalization
MNN	**M**utual **N**earest **N**eighbor
PCA	**P**rincipal **C**omponent **A**nalysis
PCK	**P**ercentage of **C**orrect **K**eypoints
PnP	**P**erspective-**n**-**P**oint
POI	**P**oint **O**f **I**nterest

LIST OF ABBREVIATIONS

RANSAC	**RAN**dom **SA**mple **C**onsensus
RMSE	**R**oot **M**ean **S**quare **E**rror
RR	**R**elative **R**esponse
RPE	**R**elative **P**ose **E**rror
RRE	**R**elative **R**otation **E**rror
RTE	**R**elative **T**ranslation **E**rror
SD	**S**parse **D**epth
SF	**S**parse **F**low
SfM	**S**tructure **f**rom **M**otion
SfS	**S**hape **f**rom **S**hading
SGD	**S**tochastic **G**radient **D**ecent
SIFT	**S**cale-**I**nvariant **F**eature **T**ransform
SLAM	**S**imultaneous **L**ocalization **A**nd **M**apping
SN	**S**witchable **N**ormalization
SSIM	**S**tructural **S**imilarity **I**ndex **M**easure
SSN	**S**parse **S**witchable **N**ormalization
VO	**V**isual **O**dometry

Chapter 1

Introduction: the Case of Quantitative Endoscopy

In this thesis, our goal is to provide the vision-based methods for applications related to quantitative endoscopy, where we try to obtain quantitative information from only an endoscopic video with the purpose of intra-operative assistance and post-operative analysis. In this chapter, we first introduce the history of endoscopy in terms of instrumentation and procedure. Next, we describe the challenges in endoscopy that we confront and the opportunities that we work towards in this thesis. We then describe the thesis statement, list the main contributions of the thesis, and give an overall diagram on how each chapter relates to others.

CHAPTER 1

1.1 Endoscopy

Endoscopy is a technique allowing inspection, manipulation, and treatment of internal organs using devices to enhance visualization from a distance of the target organs without the need of an incision large enough to allow the hand or fingers of the surgeon to enter the surgical field [1].

1.1.1 The History of Endoscopy

Endoscopy has a history of thousands of years. Endoscopic-like tools and practices have been discovered in Egypt as far back as 1700-1600 BCE [2]. The concept of minimal invasiveness was revisited by Hippocrates II from 460-375 BCE and he was very influential in advocating minimal surgical intervention as a medical practice [3]. After the early inventions in Egypt and Greece, the Romans also began using endoscopic techniques and instruments in the first century CE. However, because of the lack of technical development in visibility and accessibility, only within the last hundred years or so, has the endoscopy been mature enough to be used for practical inspection and guided surgery.

Philip Bozzini, an Italian-German physician, is considered the inventor of the field of endoscopy [2] because he developed a simple tubular device with candlelight and mirrors for illumination, in 1806, to inspect internal structures of patients. A French

physician named Antonin Jean Desormeaux was considered to have invented the first effective endoscope in 1843 [4]. Although the camera was developed through the ages as a stand-alone device, it was not utilized in combination with endoscopy until 1858 when Johann Czermak incorporated the two and took the first endoscopic image [2]. The first practicable esophagoscope is considered to have been designed by Johann Mikulicz in 1881, where a galvanized wire light source is used for illumination; in 1888, he also designed the first endoscope with a miniaturized electrical light bulb, inspired by the invention from Edison in 1880 [1]. Before the 20th century, the basics of an endoscope were in place, which are illumination, lens, the ability to treat and remove tissue, and the ability to document findings with images. However, many improvements are still needed before endoscopy becomes a general and broadly applicable technology.

In 1901, German Georg Kelling performed the first laparoscopy, which is on a dog, and seemingly used the same method on a few patients. He used a flexible gastroscope and advocated patient preparation including purging to reduce complications [1]. At about the same time, Hand Christian Jacobaeus, a physician in Stockholm, performed a large number of laparoscopies on humans. In 1924, CO_2 was first used for insufflation, which is still the standard for today, instead of atmospheric air by Richard Zolikofer of Switzerland, which has the advantage of spontaneous resorption and decreased chance for fire or explosion. In 1932, Rudolf Schindler and Georg

Wolf designed a series of semi-rigid instruments to reduce the risk of perforations in gastroscopy [1].

In the 1960s, several important developments took place. British scientist Harold Hopkins and German instrument engineer Karl Storz completely transformed the field of rigid laparoscopy and created the foundation for modern laparoscopic technology and surgery. In 1967, they developed the combination of a rod-lens optical system and a fiber optics bundle for cold light illumination, and this created the most detailed and true color images ever seen, even with a diameter of only a few millimeters. With the advancement of the endoscope and accessory techniques, Kurt Semm in Germany was the first to perform a laparoscopic appendectomy in 1980, and Erich Muhe was the first to perform laparoscopic cystectomy in 1985. In terms of the flexible endoscope, the first truly functional one is created by Basil Hirschowitz and made commercially available in 1960 by ACMI [5]. In the late 1960s and early 1970s, more companies start producing flexible endoscopes and the first set of inspections and surgeries were done for the colon and the airway.

For flexible endoscopy, the most important breakthrough was probably the invention of charged-coupled device by AT&T Bell Labs in 1969. In 1983, the first charge-coupled device (CCD) based video endoscope was introduced by American medical instrument manufacturer, Welch Allyn. This endoscope does not need a coherent bundle of fibers to transmit the light from the tip to the end of an endoscope. Instead,

the CCD sensor at the tip directly converts the optical image into digital signals that can be transferred via the shaft of the endoscope to the processing and display devices outside. Such a design removes all issues in the original instrument, such as poor image, fiber breakage, large diameter, *etc*.; it also provides more room for other functions within the endoscope shaft and makes more extreme tip deflections possible. The digital signals can be stored in the form of a video and displayed on a device outside and therefore can be watched by many people instead of only the endoscopist, which provides numerous benefits, such as better documentation, easier coordination, better ergonomics, *etc*. As CCD technology improving, the CCD sensor on the endoscope gets smaller and produces images with a higher resolution and capture rate. Thanks to this technology improvement, nowadays, all endoscope manufacturers provide endoscopes that produce high-definition images.

In the last decade of the 20th century, laparoscopic surgery took off with the help of the video endoscope and advanced accessories (*e.g*., laparoscopic clip applier). Nowadays, nearly all types of organ resections can be performed using laparoscopic techniques. Many open surgical procedures have been replaced with laparoscopic counterparts, which leads to equal or better long-term outcomes, lower patient morbidity, shorter hospital admission duration, and shorter patient recovery times.

By the early 21st century, most hollow, not blood-filled human organs have been routinely inspected using endoscopes. One exception is small bowel, which is time-

consuming, requiring hours of scope advancement, to reach with a flexible endoscope. Scientists from Israel and the United Kingdom developed a miniature endoscope in the shape of a large capsule. The patient can swallow the capsule and the images will be transmitted wirelessly to the receivers outside. As the technique of capsule endoscopes evolves, some products have multiple CCDs and do not need a receiving device, where all images are stored locally in the capsule and can be retrieved after the anal passage.

Robotics has also been introduced to endoscopy around the same time. In 1994, the first robotic surgical equipment was approved by FDA, which was used to move an endoscope inside the patient's body with voice commands. In 2000, the first system for general robotic surgery became FDA approved. Nowadays, a robotic system allows surgeons to operate on the patient's body with enhanced vision and much greater precision and control than the human hand. Such a system has been applied to many surgical areas, such as cardiac, colorectal, general, gynecologic, head&neck, brain, thoracic, and urologic surgery. The continuous improvement of accessory technologies and endoscopes also enables the rapid growth in the number and complexity of endoscopic procedures.

1.1.2 Towards Quantitative Endoscopy

As of today, endoscopy has been applied to inspect almost all the anatomy of a human where the endoscope can reach and observe, and many diseases can be treated using endoscopy with specialized instrumentation. The benefits of smaller or no scars, less morbidity, and quicker recovery are widely accepted, compared with open methods in procedural medicine and surgery. This leads to endoscopy completely or partially replacing the roles of open methods in many areas.

Nowadays, most operations are still done with qualitative assessment from the endoscopists, such as mentally aligning the orientation of the endoscope with respect to the patient anatomy and memorizing critical structures under recognizable anatomical landmarks. With the advancement of the endoscope (higher-quality images) and computing device (higher processing power), vision-based algorithms can be applied in real-time during a procedure and large-scale video analysis can be conducted, where quantitative information can be extracted. This potentially opens up many opportunities for endoscopy to further improve and produce an even larger impact. Many challenges existing today for endoscopy are also due to the lack of quantitative information and can be mitigated with that being available.

Computer vision in endoscopy is a relatively new field with many open research questions, partially because there are unique challenges to apply vision-based methods to endoscopic videos, such as illumination variation, scarce and repetitive tex-

tures, and tissue deformation. In the rest of this section, we present several examples in terms of challenges and opportunities of endoscopy that are related to this thesis.

With the field of endoscopy expanding quickly, it is becoming impossible for a single endoscopist to master all aspects of endoscopy. The educational resources, such as experts and clinical cases, can often be limited with the rapidly growing number of students who desire to be trained to perform endoscopic procedures that have increasingly more categories and larger complexity. Therefore, developing a simulation environment to train and retrain endoscopists in all aspects of procedures, such as pre-procedure planning, interprocedural communications, and management of complications, is becoming an essential component of endoscopy.

Training with simulation usually involves a 3D patient anatomy model to operate on. Several branches of methods are available to build such a simulation environment, which are mechanical, in-vivo, ex-vivo, computerized methods. Computerized methods have a larger variety in terms of types of procedures and interventions compared with other branches. The anatomical models used as the simulation environment are often obtained from Computed Tomography (CT) scans, and some further map tissue textures to the models manually, which is time-consuming and may not provide realistic appearance information [6]. If a photo-realistic textured 3D surface model of the anatomy can be estimated directly from an endoscopic video, given a large number of such videos available, the simulation models will have a much larger

diversity and appear closer to the actual scene observed by an endoscope. Besides, if endoscope trajectories can be estimated from the videos operated by expert endoscopists, these can provide valuable guidance to demonstrate what a scanning path should be for specific patients based on the tissue appearance and geometry of the anatomy. However, previous computer vision methods are not advanced and robust enough to produce an accurate surface model and endoscope trajectory from a monocular endoscopic video, which is one of the motivations of this thesis and what we try to address in Chapter 2 and 3.

Though the technologies have been advanced enough so that endoscopists can visualize inner anatomies clearly and easily, endoscopy is still mostly an operator-dependent technology. The quality of an endoscopic procedure is directly related to the attitude and level of skills of the person who drives the endoscope [1]. One example is colonoscopy, where an endoscopist looks into the colon to search for and remove polyps, which most colorectal cancers start as [7], as thorough as possible. As evaluated in Hong et al. [8], 23% of the colon surface is missing during a virtual colonoscopy in a simulation environment. As this is also the case for actual colonoscopy, the risk of developing colorectal cancer can be decreased if all polyps are detected and removed during the scoping procedure. For this reason, being able to quantitatively track the motion of the endoscope and reconstruct a surface model of the observed anatomy intra-operatively is crucial. Simultaneous Localization and Mapping (SLAM) is a

suitable type of algorithm for this purpose. However, current SLAM systems have difficulty in performing robustly and accurately in endoscopic videos because of challenges such as scarce textures and illumination changes. In Chapter 5, we develop a dense SLAM system with deep representation to confront these challenges.

When inspection or surgeries are performed under the endoscope, there is a risk of iatrogenic perforations [9]. In cases where critical structures underneath the surface get damaged, the consequence can be detrimental. For example, endoscopic endonasal surgery has become the surgical treatment of choice in many patients who require sinonasal or anterior skull base surgery. Endoscopic Endonasal Surgery (ESS) requires a thorough knowledge of anatomy, in particular, the relationship of the nose and sinuses to adjacent vulnerable structures such as the orbit or base of the skull. However, malformations, previous operations, and massive polyposis may interfere greatly with the intra-operative orientation of surgeons and this leads to major risks for the patients. Major surgical risks in EES include partial loss of vision or blindness, diplopia, damage to the cribriform plate or the roof of the ethmoid sinuses, and injury to the internal carotid artery in the wall of sphenoid sinus [10]. Therefore, having a surgical navigation system that, in real-time, tracks the endoscope and shows the spatial relationship between the scope and the surrounding anatomy can greatly reduce the risk.

Many surgical navigation systems have been commercialized (*e.g.*, LandmarX), but

most of these are landmark-based methods and therefore rely on marker-based pre-operative and intro-operative registration to align the scope trajectory with the pre-operative imaging, such as Computed Tomography. In cases of tissue deformation and manipulation during the surgery, the accuracy can largely degrade because no visual information observed by the endoscope is used for registration during the operation. Also, setting up such a system is time-consuming and the original protocol of the procedures is changed, which is not desired by many surgeons. To this end, we design a dense SLAM system in Chapter 5 with an automatic video-CT registration method in Chapter 4.

Besides the challenges in the endoscopic procedures, the visual information from the endoscopic video itself can also be exploited quantitatively for post-operative analysis. Many diseases are defined by aberrations in human geometry, such as laryngotracheal stenosis, obstructive sleep apnea, and nasal obstruction in the head and neck region. In these diseases, patients suffer significantly due to the narrowing of the airway. While billions of dollars are spent to manage these patients, the outcomes are not exclusively satisfactory. An example: the two most common surgeries for nasal obstruction, septoplasty, and turbinate reduction, are generally reported to, on average, significantly improve disease-specific quality of life, but evidence suggests that these improvements are short-term in more than 40% of cases [11, 12]. Some hypotheses attribute the low success rate to anatomical geometry but there are no objective

measures to support these claims. The ability to analyze longitudinal geometric data from a large population will potentially help to better understand the relationship between certain anatomy and surgical outcomes.

In current practice, CT is the gold standard for obtaining accurate 3D information about patient anatomy. However, due to its high cost and use of ionizing radiation, CT scanning is not suitable for longitudinal monitoring of such information. On the other hand, endoscopy is routinely performed in outpatient and clinic settings to qualitatively assess treatment effect, and thus constitutes an ideal modality to collect longitudinal data. To use the endoscopic video data to model the patient's surface anatomy and conduct analysis, a 3D video reconstruction method is required, which is the task that Chapter 3 works on.

There are many other exciting opportunities, which we do not work on in this thesis, in endoscopy for future research. For example, a robotic system that helps stabilize the undesired hand tremors during endoscopic procedures to assist the surgeon in operating more accurately and reduce the risk of perforation [13]; a disposable endoscope that minimizes the risk of cross-infection; an ultrathin endoscope that is safer and more cost-effective [14]; procedural automation with a robotic system to increase the consistency of surgical outcomes and reduce the workload of endoscopists and surgeons; a miniaturized and intelligent capsule robot that can perform automatic inspection and treatment when going through the small bowel of a patient [1].

1.2 Thesis Overview

1.2.1 Statement

The thesis statement is *A novel combination of computational neural networks and non-linear optimization algorithms can estimate surface geometry of anatomy, with cross-modal alignment, and endoscope trajectory from monocular endoscopic video sequences with sufficient performance to enable practical clinical applications.*

1.2.2 Contributions

The contributions of this thesis are as follows:

- A retrospective method is introduced to estimate high-accuracy sparse reconstruction and camera trajectory from a monocular endoscopic video [15]. This work demonstrates the superior performance of a learning-based dense descriptor in multi-view reconstruction under texture-scarce scenarios, compared with common local descriptors (Chapter 2).

- A retrospective method is developed for building an accurate textured surface reconstruction from a monocular endoscopic video [16, 17], where the first self-supervised learning scheme for monocular depth estimation is developed and superior performance is observed compared with a state-of-the-art multi-view

reconstruction method (Chapter 3).

- A multi-modal global point cloud registration method is proposed for automatic alignment between samples from different imaging modalities [18]. Specifically, a novel network normalization method is developed that shows to be more robust to task-irrelevant mean-std variation than common normalization techniques (Chapter 4).

- A real-time SLAM system is developed that robustly tracks the endoscope and produces dense surface geometry from a monocular endoscopic video stream. Learning-based appearance and optimizable geometric representations are used during SLAM system run to achieve superior performance in the texture-scarce environment, compared with a state-of-the-art feature-based SLAM system (Chapter 5).

In summary, we develop a pipeline, in both retrospective and online manners, that can accurately reconstruct surface geometry of anatomy and endoscope trajectory from a monocular endoscopic video, with automatic model alignment across different imaging modalities. With the works in this thesis, many valuable applications can potentially be enabled with the requirement of only a monocular endoscopic video, such as large-scale endoscope trajectory analysis, clinic-related anatomical geometry analysis, surgical navigation, intelligent endoscope holder, and automatic endoscopy

inspection.

In the field of endoscopy, many difficulties exist, such as illumination variation, scarce and repetitive textures, etc., and prevent prior vision-based methods from working properly. In this thesis, confronting these challenges, we combine the deep learning approaches and the non-linear optimization algorithms in novel ways. This hybrid approach brings the best of the two worlds together by effectively exploiting both the high expressivity of the former and the high accuracy of the latter. Ultimately, this allows us to make the applications above feasible with the desired level of accuracy and robustness.

1.2.3 Outline

To demonstrate the relationship between different chapters, we arrange the components from chapters in the form of a system diagram that handles the task of surface reconstruction and trajectory estimation with automatic video-CT alignment. The thesis outline is shown in Fig. 1.1. There are two branches of surface reconstruction generation in this figure. One is the retrospective one, where the input video is first input to the module in Chapter 2 to produce a point cloud and camera trajectory estimate. These estimates are then input to the module in Chapter 3 to generate a textured surface reconstruction, which, after that, is fed to the module in Chapter 4 to align the reconstruction with the CT model. Another branch is the real-time SLAM

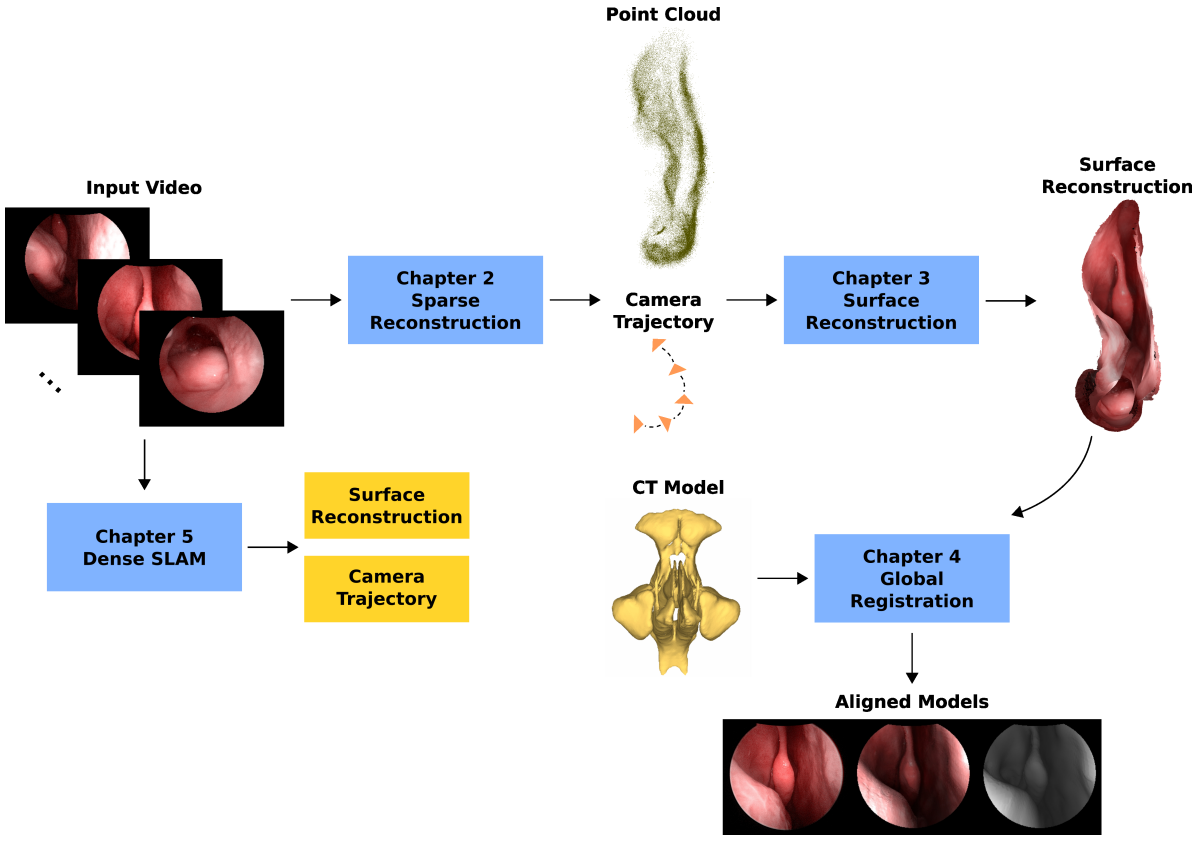

Figure 1.1: **Diagram of thesis outline.** Note that the output from the module in Chapter 4, shown in the figure above, is a side-by-side visualization of a registered video frame, surface reconstruction, and CT model.

system in Chapter 5, which accepts a video stream as input and estimates camera pose and dense geometry per keyframe. The results can then be fused into a surface reconstruction with the same technique described in Chapter 3.

Chapter 2

Sparse Reconstruction with Deep Image Features

Being able to obtain a quantitatively accurate sparse reconstruction of surface anatomy and camera trajectory is an important step towards quantitative endoscopy. This target task is already valuable for many applications, such as trajectory analysis of endoscopy procedures on a large population. Structure from Motion (SfM) is a type of algorithm that can jointly estimate sparse 3D reconstruction of the environment and the camera poses related to the 2D measurements [19]. SfM is known to be able to estimate reconstruction and poses accurately and retrospectively, and many such methods have been developed for general scenes [19–24] and clinical procedures such

Part of the materials in this chapter are from Liu *et al*. [15].

as endoscopy [25–30]. In the field of endoscopy, SfM is still an open research field because of several challenges, such as texture scarceness and the dynamic environment caused by periodic tissue deformation or surgical manipulation. In this chapter, we introduce a retrospective method to estimate high-accuracy sparse reconstruction and camera trajectory from a monocular endoscopic video. Specifically, to make SfM better suited to the texture-scarce scenario, we exploit deep image features to improve the accuracy of pair-wise feature matching commonly used in an SfM pipeline. In our evaluation, we demonstrate that SfM with the deep image features can produce denser reconstructions and more complete camera trajectories, compared with the previous hand-crafted and learning-based feature descriptors.

2.1 Related Work

2.1.1 Image Feature Descriptor

Image feature descriptors are often used to establish 2D point correspondences between two images. A local descriptor usually consists of a feature vector computed from an image patch whose size and orientation are determined by a keypoint detector, such as Harris [31], FAST [32], and DoG [33]. The hand-crafted local descriptor, scale-invariant feature transform (SIFT) [33], has been arguably the most popular

feature descriptor for correspondence estimation and related tasks. In recent years, advanced variants of SIFT have been proposed, such as RootSIFT [34], RootSIFT-PCA [35], and DSP-SIFT [36]. Some of these outperform the SIFT descriptor in tasks such as fundamental matrix estimation [37], pair-wise feature matching, and multi-view reconstruction [38]. Additionally, learning-based local descriptors have grown in popularity with the advent of deep learning, with recent examples being L2-Net [39], GeoDesc [40], and HardNet [41]. Though learning-based methods have outperformed hand-crafted ones in many areas of computer vision, advanced variants of SIFT continue to perform on par with or better than their learning-based local descriptors [37, 38]. Local descriptors generally have worse performance than dense ones, which are described later, because of the smaller context information used and the reliance on repeatable keypoint detection across images.

Several dense descriptors have been proposed, such as DAISY [42], UCN [43], and POINT^2 [44]. Compared with local descriptors, which follow a *detect-and-describe* approach [45], dense descriptors do not use a keypoint detector to find specific locations for feature extraction and instead compute features densely. As a result, dense descriptors have higher computation efficiency than local descriptors in applications that require dense matching. This way of feature extraction also walks around the challenge of repeated keypoint detection [45] across images. Learning-based dense descriptors typically show better performance compared with hand-crafted ones. This

is because CNN can encode and fuse high-level context and low-level texture information more effectively than manual rules given enough training data. Our proposed image feature descriptor belongs to the category of learning-based dense descriptors. Some works jointly learn a dense descriptor and a keypoint detector, such as SuperPoint [46] and D2-Net [45], or learn a keypoint detector that improves the performance of a local descriptor, such as GLAMpoints [47].

2.1.2 Sparse Reconstruction in Endoscopy

In the field of endoscopy, researchers have applied SfM and SLAM to video from various anatomy, including sinus [48], stomach [27–30], abdomen [49, 50], and oral cavity [51]. Popular SfM pipelines, such as COLMAP [52], and SLAM systems, such as ORB-SLAM [53, 54], do not often achieve satisfactory results in endoscopy without further improvement. Several challenges stand in the way of successful sparse reconstruction from an endoscopic video. First, tissue deformation, as in video from a colonoscopy, violates the static scene assumption in these pipelines. To mitigate this issue, researchers have proposed SfM and SLAM-based methods that allow scene deformation [23, 55–58]. Second, the textures in endoscopy are often smooth and repetitive, which makes the image feature matching error-prone. Widya *et al*. [28, 29] proposed spreading IC dye in the stomach to manually add texture to the surface. This improves the matching performance of feature descriptors and thus leads to

denser and more complete reconstructions. Qiu *et al*. [51] use a laser projector to project patterns on the surface of the oral cavity to add more textures to improve the performance of a SLAM system. However, introducing additional procedures as above is usually not desired by surgeons because the original workflow will be interrupted. In this work, we develop a dense descriptor that works robustly on the texture-scarce surface to replace the original local descriptors in these sparse reconstruction systems.

2.2 Contributions

Our contributions are as follows:

- To our knowledge, this is the first work that applies learning-based dense descriptors to the task of multi-view reconstruction in endoscopy. A learning-based dense descriptor does not need repeatable keypoint detections across images and is highly expressive in both global and local context encoding. This makes it more suitable for multi-view reconstruction under the texture-scarce scenario that is commonly seen in endoscopy, compared with local hand-crafted feature descriptors.

- We present an effective self-supervised training scheme that includes a novel loss called Relative Response Loss that can train a high-precision dense de-

scriptor with the learning style of keypoint localization. The proposed training scheme outperforms the popular hard negative mining strategy used in various learning-based descriptors [41, 43, 59].

From the system point of view, this work enables SfM to produce sparse 3D reconstructions with higher point density and more complete and accurate camera trajectory estimates. Specifically, regarding the data dependency on the system design of this thesis, the obtained information in this work will be used for depth network training and depth fusion in Chapter 3. The surface models from Chapter 3 will be used for geometric feature learning in Chapter 4. The camera trajectory estimates from Chapter 2 and surface models from Chapter 3 will be further used jointly for representation learning in Chapter 5.

2.3 Structure from Motion

We introduce a typical design of an incremental SfM algorithm as background knowledge in this section. SfM is a technique that reconstructs the 3D structure of the environment from a series of 2D observations taken from different viewpoints. Incremental SfM (denoted as SfM in this section) is a sequential processing pipeline that reconstructs the environment iteratively. In general, SfM consists of two stages, 2D correspondence search on image pairs, and incremental optimization of 3D recon-

struction and camera poses with bundle adjustment (BA) [60].

2.3.1 Correspondence Search

From a set of input images $\mathcal{I} = \{\boldsymbol{I}_i | i = 1, \ldots, N_I\}$, the image pairs with scene overlap are first obtained. On each image of a pair, 2D correspondences of the same 3D point will be identified.

Feature Extraction. For each image I_i, a set of local features $\mathcal{F}_i = \{(\boldsymbol{x}_j, \boldsymbol{f}_j) \, | j = 1, \ldots, N_{F_i}\}$ will be extracted at locations $\boldsymbol{x}_j \in \mathbb{R}^2$ represented by feature descriptions $\boldsymbol{f}_j$. A typical local feature descriptor to obtain $\mathcal{F}_i$ is SIFT [33], which first detects keypoint locations on an image and then compute feature descriptions from the patches centered at those detected locations.

Matching. For image feature matching, SfM discovers images with scene overlap and establishes 2D point correspondences between all image pairs. The simplest approach is to search for correspondences in all combinations of two images. For each feature description in the source image I_a, the most similar feature in the target image is obtained as the matched point. This approach, however, is computationally expensive with complexity $O\{\left(N_I^2 N_{F_i}^2\right)$. Various approaches try to tackle this problem [61–64]. The output of this stage is a set of image pairs, $\mathcal{C} = \{\{\boldsymbol{I}_a, \boldsymbol{I}_b\} | \boldsymbol{I}_a, \boldsymbol{I}_b \in \mathcal{I}, a < b\}$, with potential scene overlap, and the associated feature correspondences $\mathcal{M}_{ab} \in \mathcal{F}_a \times \mathcal{F}_b$.

Geometric Verification. The correspondences established in the previous stage are based only on image appearances and this stage is used to verify that the image pairs indeed have scene overlap and remove outlier feature matches within every valid image pair. SfM verifies the feature matches by estimating a transformation that maps the points in one image to another. Depending on the type of environment, different transformations can be estimated. For a purely rotating or moving camera capturing a planar scene, a homography can be used. Epipolar geometry describes the relation between two views through an essential matrix, with known camera intrinsics, or a fundamental matrix otherwise. A trifocal tensor can also be used to describe the relationship among three views [65]. To estimate a transformation, a robust estimation method, such as RANSAC [66], is often used because the feature matches from the last stage often have a large percentage of outliers. With the estimated transformation, if a sufficient number of feature points can be mapped to the matched points in the other image, the image pairs will be considered geometrically verified and the feature match outliers will be removed. The output of this stage is a scene graph with images as nodes and verified image pairs as edges.

2.3.2 Incremental Reconstruction

With the above scene graph as input, the reconstruction pipeline here will produce a set of pose estimates $\mathcal{P} = \{\boldsymbol{T}_c \in \mathrm{SE}(3)\}$ for successfully registered images, and the sparse 3D reconstruction as a set of points $\mathcal{X} = \{\boldsymbol{p}_k \in \mathbb{R}^3 | k = 1, \ldots, N_X\}$.

Initialization. Incremental reconstruction initializes first with a two-view reconstruction [67]. The quality of the initial pair is important because the reconstruction may never recover from a bad initialization. The choice of initialization will also affect the final SfM performance in terms of robustness, accuracy, and speed. Typically, initializing from a dense location in the image graph with many overlapping cameras results in a more robust and accurate reconstruction. Initializing from a sparser location, the runtime will be lower.

Image Registration New images can be registered to the current model by solving the Perspective-n-Point (PnP) problem [68]. The 2D-3D correspondences used in PnP will be the ones between the 2D feature points in the new images and the triangulated 3D points in the already registered images. The PnP problem involves estimating the pose $\boldsymbol{T}_c$ and intrinsic parameters if the camera is uncalibrated. Because the 2D-3D correspondences are often outlier-contaminated, a robust solver needs to be used for optimization [69].

Triangulation An image will be registered only if some of the existing scene points are visible in the image. When a new image gets registered, a set of new scene

points could also be triangulated as long as these points are visible in at least one more image from a different viewpoint. Triangulation is a crucial step in SfM, as it increases the stability of the existing model through redundancy [70]. It also provides additional 2D and 3D information to enable registering new images. Many methods exist for multi-view triangulation [71–73].

Bundle Adjustment In the triangulation stage, image registration and triangulation are separate procedures. However, these two processes are highly correlated and the optimal solution can only be achieved with joint optimization. Therefore, without joint refinement, SfM with only image registration and triangulation often have large drifting errors. BA [70] is a non-linear optimization process for jointly refining camera parameters $\boldsymbol{T}_c$ and scene points $\boldsymbol{p}_k$. The objective to optimize over is often the reprojection error

$$\mathcal{E} = \sum_j \rho\left(\|\pi\left(\boldsymbol{T}_c, \boldsymbol{p}_k\right) - \boldsymbol{x}_j\|_2^2\right) \quad , \tag{2.1}$$

where the function π projects scene points to image space. Because the correspondences often contain outliers, an outlier-robust loss, such as Huber [74], can be used for the loss function ρ. A non-linear optimization solver is needed to minimize the objective above by jointly optimizing the camera parameters and scene points. Levenberg-Marquardt (LM) [75] is a suitable choice for this purpose. With all the

stages described in the incremental reconstruction section completed or converged for the input scene graph, SfM will be considered finished.

2.4 Learning-based Features for Image Matching

Replacing the role of a hand-crafted local descriptor (*e.g.*, SIFT), the proposed learning-based feature is used in the *Feature Extraction* stage of the described SfM pipeline above. It improves the SfM performance by increasing the accuracy of feature matching.

2.4.1 Network Architecture

As shown in Fig. 2.1, the training network is a two-branch Siamese network. The input is a pair of color images, which are used as source and target. The training goal is, given a keypoint location in the source image, to find the correct corresponding keypoint location in the target image. An SfM method [48] with SIFT is applied to video sequences to estimate the sparse 3D reconstructions and camera poses. The groundtruth 2D point correspondences are then generated by projecting the sparse 3D reconstructions onto the image planes using the estimated camera poses. The

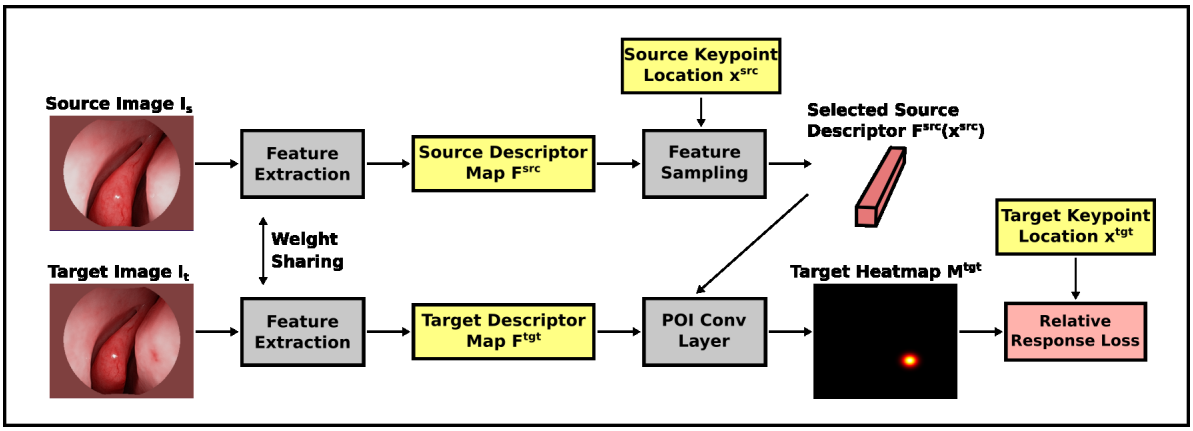

Figure 2.1: **Diagram of image feature descriptor learning.** The training data consists of a pair of source and target images and groundtruth source-target 2D point correspondences. The source and target images are randomly selected from the frames which share observations of the same 3D points. For each pair of images, a certain number of point correspondences are randomly selected from the available ones in each training iteration. For the simplicity of illustration, only one target-source point pair and the corresponding target heatmap are shown in the figure. All concepts in the figure are defined in Sec. 2.4. © 2020 IEEE

dense feature extraction module is a fully convolutional DenseNet [76] which takes in a color image and outputs a dense descriptor map that has the same resolution as the input image and the length of the feature descriptor as the channel dimension. The descriptor map is L2-normalized along the channel dimension to increase the generalizability [77].

For each source keypoint location, the corresponding descriptor is sampled from the source descriptor map. Using the descriptor of the source keypoint as a 1×1 convolution kernel, a 2D convolution is performed on the target descriptor map in the POI Conv Layer [44], which converts the problem of descriptor learning to keypoint

localization. The computed heatmap represents the similarity between the source keypoint location and every location on the target image. The network is trained with the proposed Relative Response (RR) Loss [15] to force the heatmap to present a high response only at the groundtruth target location.

2.4.2 Loss Design

We use RR Loss to train the image feature descriptor. The RR Loss is proposed with the intuition that a target heatmap should present a high response at the groundtruth target keypoint location and the responses at other locations should be suppressed as much as possible. Besides, no prior knowledge is desired to be assumed on the response distribution of the heatmap, which preserves the potential of multimodal distribution and respects the matching ambiguity of challenging cases. To this end, we propose to maximize the ratio between the response at the groundtruth location and the summation of all responses of the heatmap. For a pair of source and target input images $\boldsymbol{I}^{\mathrm{src}}, \boldsymbol{I}^{\mathrm{tgt}} \in \mathbb{R}^{3\times H\times W}$, a pair of dense descriptor maps, $\boldsymbol{F}^{\mathrm{src}}, \boldsymbol{F}^{\mathrm{tgt}} \in \mathbb{R}^{C\times H\times W}$, are generated from the feature extraction module. For a descriptor at the source keypoint location $\boldsymbol{x}^{\mathrm{src}} \in \mathbb{R}^2$, the corresponding feature descriptor, $\boldsymbol{F}^{\mathrm{src}}(\boldsymbol{x}_{\mathrm{s}}) \in \mathbb{R}^{C\times 1\times 1}$, is extracted with the nearest neighbor sampling, which could be changed to other sampling methods if needed. By treating the sampled feature descriptor as a convolution kernel, the 2D convolution operation is performed on $\boldsymbol{F}^{\mathrm{tgt}}$

to generate a target heatmap $\boldsymbol{M}^{\mathrm{tgt}} \in \mathbb{R}^{1\times H\times W}$, which stores the similarity between the sampled source descriptor and every target descriptor in $\boldsymbol{F}^{\mathrm{tgt}}$. Mathematically, the RR loss is defined as,

$$\mathcal{L}_{\mathrm{rr}} = -\log\left(\frac{\mathrm{e}^{\sigma \boldsymbol{M}^{\mathrm{tgt}}\left(\boldsymbol{x}^{\mathrm{tgt}}\right)}}{\sum_{\boldsymbol{x}\in\Omega}\mathrm{e}^{\sigma \boldsymbol{M}^{\mathrm{tgt}}(\boldsymbol{x})}}\right) \quad , \tag{2.2}$$

where $\sigma \in \mathbb{R}$ is applied to enlarge the value range of the heatmap $\boldsymbol{M}^{\mathrm{tgt}}$, which becomes $[-\sigma, \sigma]$. A spatial softmax is then calculated at the groundtruth location $\boldsymbol{x}^{\mathrm{tgt}}$ of the scaled heatmap, where the denominator is the summation of elements of the scaled heatmap within the valid region Ω. The logarithm operation is used to speed up the convergence.

We observe that, by only penalizing the value at the groundtruth location after spatial softmax operation, the network learns to reduce the response at the other locations and increase the response at the groundtruth location effectively. We compare the feature matching and SfM performance of dense descriptors trained with different common loss designs that are originally for the task of keypoint localization in Sec. 2.5. A qualitative comparison of target heatmaps generated by different dense descriptors is shown in Fig. 2.2.

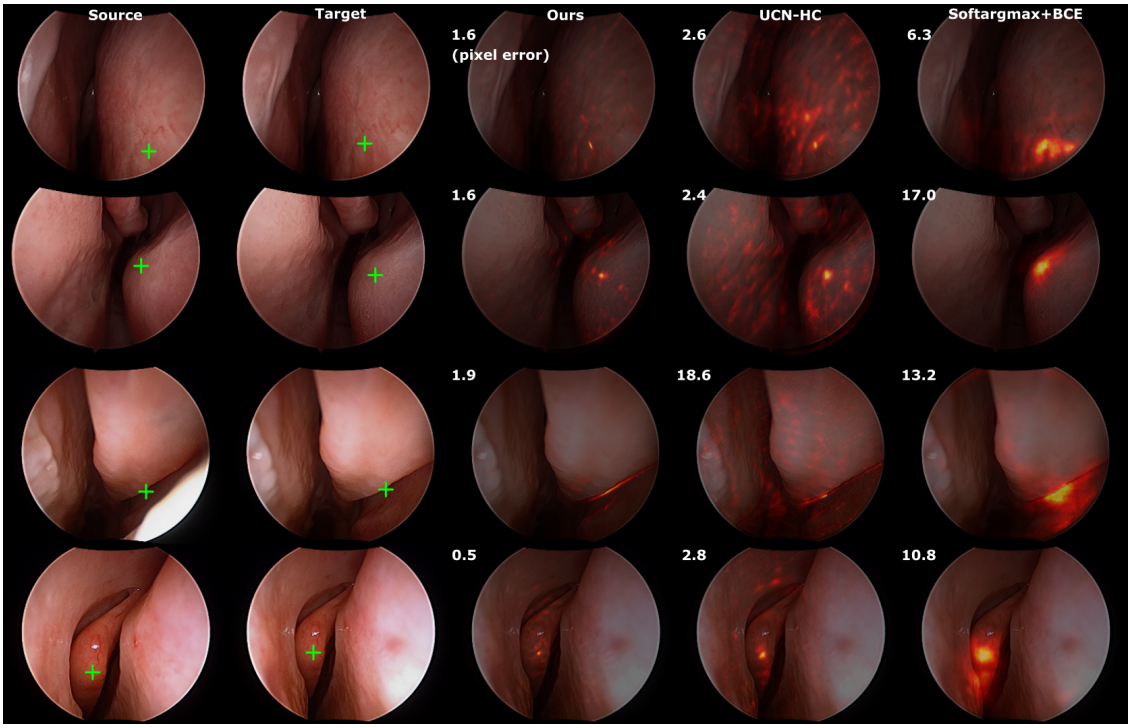

Figure 2.2: **Qualitative comparison on feature matching performance in endoscopy.** The figure qualitatively shows the performance of three dense descriptors trained with different loss designs on the task of pair-wise feature matching. The first two rows are training images and the rest are testing ones. The first and second columns show the source-target image pairs, where the green crossmarks indicate the groundtruth source-target point correspondences. For each dense descriptor, a target heatmap, as shown in the last three columns, is generated from the POI Conv Layer. To visualize the contrast better, the displayed heatmap is normalized with spatial softmax operation and then with the maximum value of the processed heatmap. The numbers shown in the last three columns are the pixel errors between the estimated target keypoint locations and the groundtruth ones. The fourth column shows the results of UCN [43] trained with the Hardest Contrastive (HC) Loss [59] on the endoscopy dataset. The model in the fifth column is trained with the same method as ours except that the training loss is Softargmax [78] and Binary Cross Entropy (BCE) Loss [79] instead of the proposed RR Loss. The results show that our method produces fewer high responses, which leads to better matching accuracy.

2.4.3 Application in Structure from Motion

In this work, we choose SfM over other multi-view reconstruction methods such as SLAM because SfM is known to produce accurate reconstruction retrospectively. Since this work serves one role to provide information to the following works in terms of training and depth fusion, accuracy is more important. The proposed descriptor is used in SfM to replace the role of a local descriptor in the pair-wise image feature matching module of an SfM pipeline. First, candidate keypoint locations are extracted from the source image. This can be obtained either with uniform sampling or a certain keypoint detector. Then, for each source keypoint location in the source image, a corresponding target heatmap is generated with the method above. The location with the largest response value in the heatmap is selected as the estimated target keypoint location. The descriptor at the estimated target keypoint location then performs the same operation on the source descriptor map to estimate the source keypoint location.

Because of the characteristics of dense feature matching used in this work, the Mutual Nearest Neighbor criterion used in the pair-wise feature matching of a local descriptor is too strict. We relax the criterion by accepting the match as long as the estimated source keypoint location is within the vicinity of the original source keypoint location, which we call cycle consistency criterion. The computation of dense matching can be parallelized on modern graphics processing unit (GPU) by treating all sampled source descriptors as a kernel $\in \mathbb{R}^{N \times C \times 1 \times 1}$; N is the number of query

source keypoint locations used as the output channel dimension; C is used as the input channel dimension of a standard 2D convolution operation. The rest of the pipeline is kept the same as a standard SfM.

2.5 Experiments

In our published work [15], we evaluated the proposed method on three datasets. Sinus endoscopy dataset was used to evaluate the performance of local and dense descriptors on the task of pair-wise feature matching and SfM in endoscopy. KITTI Flow 2015 dataset [80] was used to evaluate the performance of dense descriptors on the task of pair-wise feature matching in natural scenes. A small-scale dataset with a collection of building photos [81] was used to evaluate the performance of local and dense descriptors on the task of SfM in natural scenes. In this thesis, we present the experiments related to the endoscopy dataset only.

2.5.1 Experiment Setup

All experiments were conducted on a workstation with 4 NVIDIA Tesla M60 GPUs, each with 8 GB memory, and the method was implemented using PyTorch [82]. The nasal endoscopy dataset consists of video data collected from eight patients and two cadavers. The overall duration is around 30 minutes. For the ease of experiments, all

images are downsampled to 256×320 pixels during both training and evaluation. For the network backbone, we use a light-weight version of Fully Convolutional DenseNet (FC-DenseNet) [76] with 32 layers and filter growth rate of 10. The length of the output descriptor is 256; the overall number of parameters is around 0.53 million. The model is trained with Stochastic Gradient Descent (SGD) [83] with the cyclic learning rate [84] within the range of $[1.0\mathrm{e}^{-4}, 1.0\mathrm{e}^{-3}]$. The scale factor σ used in the RR Loss is set to 20.0. Data from five patients and one cadaver are used for training; the other cadaver is used for validation; the remaining three patients are for evaluation. Because our evaluation focuses on the loss design, for fairness, we use the same network backbone described above for all dense descriptors to extract features. All models are trained until the performance on the validation set stops improving.

2.5.2 Pair-wise Image Matching Evaluation

The evaluation results of pair-wise feature matching are shown in Table 2.1. To measure the accuracy of feature matching, we use Percentage of Correct Keypoints (PCK). PCK measures the percentage of source keypoints whose estimated corresponding location in the target image is within a certain distance of the true location. In this work, three distance thresholds for PCK are used, which are 5, 10, and 20 pixels.

The results show that our proposed training scheme for the dense descriptor out-

	UCN-C	UCN-HC	Softarg.	Softarg.+BCE	Softmax+BCE	RR+Softarg.	RR
PCK@5px	25.5	58.8	36.5	44.6	35.4	57.9	63.0
PCK@10px	35.0	67.2	54.6	63.1	51.1	68.6	71.9
PCK@20px	47.0	74.0	73.6	77.4	66.0	78.6	80.0

Table 2.1: **Evaluation of feature matching performance in endoscopy.** This table shows the average PCK with threshold 5px, 10px, and 20px over all nine sequences from the three testing patients. The PCK is calculated on all image pairs whose interval is within 20 frames. For each pair, PCK is computed by comparing the dense matching results with the groundtruth point correspondences from SfM results. The feature matching results in each column are generated by the descriptor whose name is on the first row. From left to right, the evaluated descriptors are UCN trained with Contrastive Loss (UCN-C) [43], UCN trained with HC Loss (UCN-HC) [59], replacing the proposed RR with Softargmax [78], replacing RR with Softargmax and BCE, replacing RR with spatial softmax and BCE [85], RR and Softargmax, and the proposed training scheme with RR. The model trained with the proposed RR achieves the best average matching accuracy. © 2020 IEEE

performs competing methods for dense descriptor learning, which are Contrastive Loss in [43] and HC Loss in [59]. Besides, since we convert the problem of descriptor learning to keypoint localization, we also evaluate the performance of several loss functions used in keypoint localization by training the proposed network with these instead of RR Loss. For the proposed method, generating and matching a pair of dense descriptor maps under the current setting takes about 37ms. © 2020 IEEE

2.5.3 Structure from Motion Evaluation

To evaluate the performance of local and dense descriptors on the task of SfM in endoscopy, we use a simple SfM pipeline [48] which takes in pair-wise feature matches, uses Hierarchical Multi-Affine [86] for geometric verification, and global

bundle adjustment [87] for optimization. Pair-wise feature matches are estimated in all image pairs whose interval is within 30 frames. For all local descriptors, difference of Gaussians (DoG) [33] is used to extract keypoint locations in both source and target images for sparse feature matching with Mutual Nearest Neighbors (MNN) [88] as the matching criterion. For dense descriptors, DoG is used to extract keypoint locations in only source images and dense matching is performed on the target images for these detected candidate keypoint locations in the source images. The false matches are ruled out using the cycle consistency criterion described in Sec. 2.4.3.

Because of the texture smoothness of endoscopy, we modify the parameters of DoG so that more candidate keypoint locations can be detected. The number of layers in each octave is 8; the contrast threshold is $5.0\mathrm{e}^{-5}$; the edge threshold is 100.0; the standard deviation of the Gaussian applied to the image at the first octave is 1.1. All hand-crafted descriptors use the parameter setting recommended by the original authors. The SfM results are shown in Table 2.2. Note that we build an image patch dataset from SfM results in endoscopy using the same method as [40] to fine-tune the HardNet++ [41] for a fair comparison, which indeed has a better performance compared with the pre-trained model released by the authors.

	Seq. 1-1 (381)			Seq. 1-2 (314)			Seq. 1-3 (370)			Seq. 2-1 (455)			Seq. 2-2 (630)			Seq. 2-3 (251)			Seq. 3-1 (90)			Seq. 3-2 (1309)			Seq. 3-3 (336)		
SIFT	104	474	5.62	219	1317	5.58	113	938	5.16	119	751	5.81	295	10384	6.43	122	1896	5.38	48	435	5.09	55	953	5.51	169	2169	5.57
DSP-SIFT	149	783	5.09	235	1918	5.06	132	1228	4.78	404	6557	5.32	296	7322	5.64	167	3450	5.00	42	293	4.81	150	745	5.17	180	1180	5.18
RootSIFT-PCA	104	384	5.89	219	1004	5.67	115	661	5.11	227	821	5.82	295	10147	6.43	128	2025	5.46	50	255	5.18	217	3188	5.35	176	2450	5.62
HardNet++	180	1554	4.63	233	2162	4.81	244	3003	4.65	424	4755	4.65	534	9828	4.85	225	5727	4.56	79	610	4.66	416	4658	4.62	228	3196	4.66
UCN-C	349	13402	4.26	311	13198	4.50	248	8336	4.43	405	11935	4.13	293	8258	4.46	196	9273	3.98	77	2445	4.10	503	16166	4.29	206	3736	4.17
UCN-HC	381	15274	4.84	314	13519	4.84	352	16900	4.89	455	33299	4.67	630	45375	4.81	251	26322	4.37	86	2988	4.39	484	13394	4.39	283	11555	4.39
Softarg.	348	5966	4.74	312	7774	4.74	252	7426	4.63	293	4861	4.50	547	12590	4.24	205	2847	4.22	59	534	4.17	451	7247	4.76	302	6039	4.26
Softarg.+BCE	357	11502	4.47	314	10373	4.57	244	10339	4.55	426	19848	4.34	560	22482	4.19	125	1150	4.04	46	774	4.04	500	12187	4.51	303	6268	4.06
Softmax+BCE	165	2246	4.26	306	8885	4.26	228	8628	4.19	378	8559	4.10	296	12081	4.19	77	1124	3.96	34	353	4.02	261	5024	4.19	181	2973	4.07
RR+Softarg.	381	19921	4.99	314	20375	4.98	256	20550	4.94	455	44388	4.75	630	39752	4.64	244	10055	4.35	87	5071	4.33	507	20906	4.61	312	12856	4.36
RR	381	27317	5.07	314	22898	5.23	367	29734	5.06	455	41380	4.78	630	45654	4.80	251	19645	4.43	89	6763	4.62	507	35645	4.68	313	21703	4.53

Table 2.2: **Evaluation of SfM performance in endoscopy.** We compare the SfM results of nine sequences from the three testing patients. The SfM results are generated by the descriptors whose names are on the first column. We compare the SfM performance of local and dense descriptors. Starting from the first descriptor, these are SIFT [33], DSP-SIFT [36], RootSIFT-PCA [35], HardNet++ [41] fine-tuned with the endoscopy dataset, UCN trained with Contrastive Loss (UCN-C) [43], UCN trained with HC Loss (UCN-HC) [59], replacing the proposed RR Loss with Softargmax [78], replacing RR with Softargmax and BCE, replacing RR with spatial softmax and BCE [85], RR and Softargmax, and the proposed training scheme with RR. Each number in the first row represents the number of frames in each sequence. In the following rows, for each sequence and each method, three numbers from left to right are the number of registered views, the number of sparse points, and the average track length of sparse points. It shows that the proposed RR obtains the most number of registered views in all sequences and the densest reconstructions for most of the sequences. SIFT or RootSIFT-PCA achieve the highest average track length in all sequences.

2.6 Discussion

2.6.1 Intuition on the Descriptor Training Schemes

We attribute the performance difference between our method and UCN-HC to the different strategies of training data sampling. For UCN-HC, given a positive point pair, one hardest negative point is obtained in a minibatch for each of the points in the pair to calculate the negative loss. A diameter threshold is also set to avoid mining points that are too close to the positive point. A positive margin threshold and negative margin threshold are also set to avoid penalizing positive pairs that are close enough or negative pairs that are far enough. There are several potential problems with this setting. First, the strategy of hardest sample selection, which was also used similarly in the local descriptor training [41], could potentially lead to training instability, which was also mentioned by the original authors in their source code repository. Because for each iteration of training, only the hardest negative samples in a minibatch provide gradients to the network training with other samples being ignored, the gradient direction may not be helpful to these ignored samples. This could potentially lead to training oscillation where the hardest samples jump among different samples but the network never converges to the optimal solution. The results of instability can be found in Fig. 2.2, where many high responses are scattered in the heatmap. Second, the manually specified diameter and margin thresholds could

also lead to a suboptimal solution. Because samples that are within the diameter of a selected sample are not considered as negative ones, the network will never try to push nearby samples away from the selected one. Therefore, this limits the matching precision of the descriptor. This again can be observed in Fig. 2.2, where the high-response clusters around the groundtruth target locations appear to be wider than our proposed method. The margin thresholds in the loss design also remove the possibility of further pushing away negative samples from the positive ones and pulling positive pairs closer, which could be another reason for obtaining such heatmaps.

As a comparison, in our method, for each sampled point in the source image, all points in the target image are observed in one training iteration. Only the groundtruth target point is considered a positive point and all other points are considered negative ones. This avoids the oscillation related to the descriptor distance between the selected source point and all points in the target image. The reason why this training scheme will not suffer from the problem of data imbalance is due to the proposed RR Loss. The goal of RR Loss is to make the ratio between the response at the groundtruth target location and the summation of all responses in the target image as high as possible. By doing this, the network will try to suppress all responses except the one at the target groundtruth location. It does not assume any prior distribution of the response heatmap and conveys the goal of precise feature matching clearly, which we believe improves the expressivity of the network.

We have also evaluated some common losses used in the task of keypoint localization, such as spatial softmax + BCE and Softargmax [78]. Spatial softmax + BCE is used for heatmap regression so that the network produces a similar heatmap as the groundtruth one. However, because the groundtruth distribution is usually assumed to be Gaussian with a manually specified standard deviation, this limits the expressivity of the network in cases where Gaussian distribution is not optimal. This can be observed in the third row in Fig. 2.2, where the model trained with Softargmax + BCE tries to infer a Gaussian-like distribution around the groundtruth location. As a comparison, the learned descriptor in our proposed method naturally produces a high response along the edge of the surface, which is where the most ambiguities come from. Besides, BCE also suffers from the data imbalance problem for the case where positive and negative samples are highly unbalanced, which is also observed in [89]. Softargmax converts the task of keypoint localization to a position regression task where the network tries to produce a heatmap so that the centroid of the heatmap is close to the groundtruth target location. However, this suffers from the fact that any distribution where the centroid is equal to the target location will not be further penalized. Therefore, Softargmax makes the network easily trapped in sub-optimal solutions of learning a discriminative descriptor, whereas there are no such training ambiguities in RR Loss. Though this ambiguity can be reduced by combining Softargmax with BCE, the performance is still worse than RR Loss, as observed in

Table 2.1 and 2.2 because of the unimodal distribution assumption.

2.6.2 Local Descriptor vs. Dense Descriptor

We observe that learning-based dense descriptors usually perform better than local descriptors in the experiments related to SfM in nasal endoscopy. We attribute this to two reasons. First, local descriptors usually need a keypoint detector, such as DoG [33], to detect candidate keypoints before sparse feature matching. The lack of repeatability in the keypoint detector makes many true matches unable to be found because either source or target locations for these matches are not detected as candidate keypoints in the keypoint detection phase. As observed in [45], the unstable detection is because the detector usually uses low-level information, which is often significantly affected by changes such as viewpoint and illumination. Second, the smooth and repetitive textures in endoscopy make it challenging for the local descriptors that have a limited receptive field to find correct matches even if all points in the true matches are detected by the keypoint detector. On the other hand, learning-based dense descriptors do not rely on the keypoint detector to produce repeatable keypoint locations and have a larger receptive field.

Compared with local descriptors, dense descriptors also have disadvantages. First, a dense descriptor is more memory-demanding. This is because, to parallelize the dense matching procedure with many keypoint locations, the descriptors need to be

organized in the form described in the Sec. 2.4.3. This requires memory to store a response target heatmap for each source keypoint location before the target location is estimated from the heatmap. Though sparse matching can also be performed with a dense descriptor, the performance will degrade because of the reliance on a repeatable keypoint detector. Therefore, the practical usage of a dense descriptor on a low-cost embedded system is limited. Second, learning-based dense descriptors seem to be more overfitting-prone compared with learning-based local descriptors. This is because the dense descriptor network relies on both high-level and low-level image information to generate a descriptor map. Because high-level information, presumably, has more variation compared with low-level texture information that learning-based local descriptors only need, more training data is probably needed for a dense descriptor. The reason why dense descriptors seem to generalize well in endoscopy could be due to the lower anatomical variation compared with the variation in natural scenes.

2.7 Conclusion

In this chapter, we introduce a method to estimate accurate sparse reconstruction and endoscope trajectory from a monocular endoscopic video. An effective self-supervised training scheme with a novel loss design is proposed for the learning-based dense descriptor. To our knowledge, this is the first work that applies a learning-based

dense descriptor to endoscopy for multi-view reconstruction. We evaluate our method on an endoscopy dataset on the task of pair-wise feature matching and SfM, where our proposed method outperforms other local and dense descriptors. The comparison study helps to gain more insights on the difference between local and dense descriptors, and the effects of different loss designs on the overall performance of a dense descriptor.

This work is currently limited to the static scene because of the SfM algorithm used. In some types of endoscopy (*e.g.*, colonoscopy), however, tissue deformation is common and can be large. Therefore a sparse reconstruction method that works in a deformable environment is desired. Many non-rigid SfM works have been developed over the years [23, 55, 58, 90, 91]. By integrating the proposed image feature descriptor into one of these methods, it is feasible to make the method work in a non-rigid environment with high accuracy, density, and completeness, which could be an interesting direction to explore. In scenarios where surgical operations are applied, the tissues can be cut away. It will be interesting to see if a sparse reconstruction algorithm can ignore affected areas and only estimate the camera trajectory and sparse reconstruction based on the unchanged anatomy, which is challenging because the appearance of the unchanged part could change dramatically due to cases such as bleeding.

The dense descriptor method in this work uses self-supervised learning scheme

and therefore the method relies less on its generalizability compared with a fully-supervised one. However, from our observation, the model trained on the sinus endoscopy dataset can at least generalize across different patients and endoscopes. We would expect the trained model can also generalize to endoscopy with similar tubular structures (*e.g.*, bronchoscopy), but experiments are needed for validation.

Chapter 3

Surface Reconstruction with Deep Depth Priors

In the previous chapter, we introduced sparse reconstruction with deep image features. Though the reconstruction has high accuracy, there is no dense surface geometry obtained, which is a necessity in many applications. For example, if clinic-related geometric measurements, such as cross-section area of the nasal cavity, the volume of polyps, *etc*., need to be estimated from an endoscopic video, a watertight surface is needed. Besides, obtaining surface reconstructions from videos can enable other valuable applications, such as surgical augmented reality, statistical shape analysis, collision avoidance in robotic navigation, *etc*. However, current methods are

Part of the materials in this chapter are from Liu *et al*. [16].

either not suitable for endoscopy because of invalid assumptions (*e.g.*, photometric constancy) or inaccurate geometry (*e.g.*, incorrect topology). Volumetric depth fusion methods [92] can reconstruct a surface from depth measurements, which, however, is not readily available in a monocular endoscopic video. To this end, combining the structural priors (inductive biases) of a deep network and the accuracy of non-linear optimization, we develop a method for building an accurate surface reconstruction from a monocular endoscopic video. Specifically, we develop a self-supervised depth estimation method that can convert sparse depth signals from SfM results to dense depth estimates, by exploiting inductive biases from a convolutional neural network. These are then used in a traditional depth fusion pipeline to compute accurate surface geometry. The evaluation shows that the estimated surface reconstructions are in good agreement with CT surface models, which makes the method useful for the quantitative applications above.

3.1 Related Work

3.1.1 Depth Estimation

Deep learning has shown promising results in high-complexity problems including monocular depth estimation [93], a task that benefits from local and global context in-

formation and multi-level representations. However, training a deep learning model in a fully supervised manner with endoscopic videos is challenging because paired dense ground truth depths and endoscopic images are hard to obtain.

Several simulation-based works have been proposed to deal with this challenge by training on synthetic dense depth maps generated from patient-specific CT data. Visentini *et al*. use synthetic endoscopic videos from CT data to train a fully supervised depth estimation network. It then relies on another module to convert real video frames to the ones that have a similar appearance as the synthetic images [94]. This method requires per-endoscope photometric calibration and complex registration designed for narrow tube-like structures. In addition, it remains unclear whether this method will work on in-vivo images since validation is limited to two lung nodule phantoms. Mahmood *et al*. simulate pairs of color images and dense depth maps from CT data for depth estimation network training. During the application phase, they use a generative adversarial network (GAN) [95] to convert real endoscopic images to simulation-like ones and then feed them to the trained depth estimation network [96]. In their work, an appearance transformer network is trained separately by simply mimicking the appearance of simulated images but without knowledge of the target task, *i.e*., depth estimation, which can lead to decreased performance up to incorrect depth estimates.

Besides simulation-based methods, hardware-based solutions exist that may be

advantageous in the sense that they usually do not rely on pre-operative imaging modalities [97, 98]. However, incorporating depth or stereo cameras into endoscopes is challenging and, even if possible, these cameras may still fail to acquire dense and accurate enough depth maps from endoscopic scenes for fully supervised training because of the non-Lambertian reflectance properties of tissues and the paucity of features.

Several self-supervised approaches for monocular depth estimation have been proposed in the general field of computer vision [99–102]. Based on our observations and experiments, however, these methods are not generally applicable to endoscopy because of several reasons. First, photometric constancy between frames assumed in their work is not available in endoscopy. The camera and light source move jointly, and therefore, the appearance of the same anatomy can vary substantially with different camera poses, especially for regions close to the camera. Second, appearance-based warping loss suffers from gradient locality, as observed in [101]. This can result in network training getting trapped in bad local minima, especially for textureless regions. Compared to natural images, the overall scarcer and more homogeneous texture of tissues observed in endoscopy, *e.g.*, nasal endoscopy and colonoscopy, makes it even more difficult for the network to obtain reliable information from photometric appearance. Moreover, estimating a global scale from monocular images is inherently ambiguous [103]. In natural images, the scale can be estimated using learned prior

knowledge about sizes of common objects, but there are no such visual cues in endoscopy, especially for images where no known instruments are present. Therefore, approaches that try to jointly estimate depths and camera poses with correct global scales are unlikely to work in endoscopy.

The observations above demonstrate that recent self-supervised approaches cannot enable the network to capture long-range correlation in either spatial or temporal dimension in imaging modalities where no lighting constancy is available (*e.g.*, endoscopy). On the other hand, traditional multi-view reconstruction methods, such as SfM, are capable of explicitly capturing long-range correspondences with illumination-invariant feature descriptors (*e.g.*, SIFT [33]) and global non-linear optimization (*e.g.*, BA [70]). We argue that the estimated sparse reconstructions and camera poses from a multi-view reconstruction method are valuable and should be integrated into the network training of monocular depth estimation.

3.1.2 Surface Reconstruction in Endoscopy

Many methods for surface reconstruction from endoscopic videos have been proposed. Several SfM-based methods aim at tackling texture smoothness [27–30, 51]. They provide a sparse or dense reconstructed point cloud which is further processed with a surface reconstruction method, such as Poisson reconstruction [104]. The Poisson reconstruction does not exploit the topology of anatomy and its performance de-

pends on the quality of the sparse point cloud. Therefore, there is no guarantee that these approaches will result in reasonable surface estimates. This is especially true in the case of topologically complex structures, such as the nasal cavity shown in Fig. 3.5. Shape from Shading (SfS) methods are often combined with fusion techniques [105–107] and often require careful photometric calibration to ensure accuracy. Reconstruction with tissue deformation are handled in [56, 57, 108]. In intra-operative scenarios, SLAM-based methods [56, 57, 109, 110] are preferable as they optimize for near real-time execution. There are also learning-based methods [110, 111] taking advantage of deep learning advancements in depth and pose estimation to improve model quality.

3.2 Contributions

Our contributions are as follows:

- We propose a patient-specific learning-based method for surface reconstruction from monocular endoscopic videos.
- To our knowledge, the self-supervised monocular depth estimation method, proposed in this work, is the first work that requires only monocular endoscopic images during both training and application phases. This method makes the proposed surface reconstruction pipeline feasible because previous depth esti-

mation works require paired groundtruth color and dense depth data for training, which is impractical to obtain during clinical procedures.

- The produced surface reconstructions are in good agreement with CT. This potentially enables many quantitative endoscopy applications, such as measuring clinically relevant parameters directly from a video obtained through a routine endoscopy inspection.

From the system point of view, this work handles the task of reconstructing a quantitatively accurate surface model from a monocular endoscopic video. This surface model can be used in many clinical applications, such as augmented reality in surgical navigation, and quantitative longitudinal monitoring of anatomy connected to a natural orifice. For the specific system design of this thesis, the obtained surface reconstruction will be further used as supervisory signals for the geometric feature learning in Chapter 4 and the representation learning in Chapter 5.

3.3 Surface Reconstruction Pipeline

The goal of the proposed pipeline is to automatically reconstruct a surface model from a monocular endoscopic video. The pipeline, shown in Fig. 3.1, has three main components: 1) SfM with deep image features, described in Chapter 2; 2) depth estimation; and 3) volumetric depth fusion with surface extraction. Accurate reconstruc-

tions and camera trajectory estimates from SfM are important because these are subsequently used for two purposes: 1) providing supervisory signals for fine-tuning two learning-based modules, *i.e.*, *Feature Extraction* and *Depth Estimation*; 2) guiding the fusion procedure in the *Depth Fusion & Surface Extraction* module. *Depth Estimation* provides dense depth estimates for all video frames. This information is then aggregated over the whole video sequence in *Depth Fusion & Surface Extraction* to produce the surface reconstruction.

The two learning-based modules used in our approach, namely *Feature Extraction* and *Depth Estimation*, are both self-supervised because they can be trained on video sequences with corresponding SfM results obtained with a conventional hand-crafted feature descriptor. The method design and training strategy for *Feature Extraction* is introduced in Chapter 2; those for *Depth Estimation* are introduced in Sec. 3.4.

Before the pipeline shown in Fig. 3.1, *Feature Extraction* module should already be trained with the SfM result on the input video. Note that, this time, the SfM is applied to the input video with a hand-crafted feature descriptor to produce such results. Then the pair-wise feature matches from *Feature Extraction* will enable SfM to produce a denser reconstruction and a more complete camera trajectory. Please refer to Chapter 2 regarding how to produce SfM estimates with the *Feature Extraction* module. If the SfM result is unsatisfactory, the *Feature Extraction* module will be fine-tuned with this new SfM result for bootstrapping. This process can be repeated

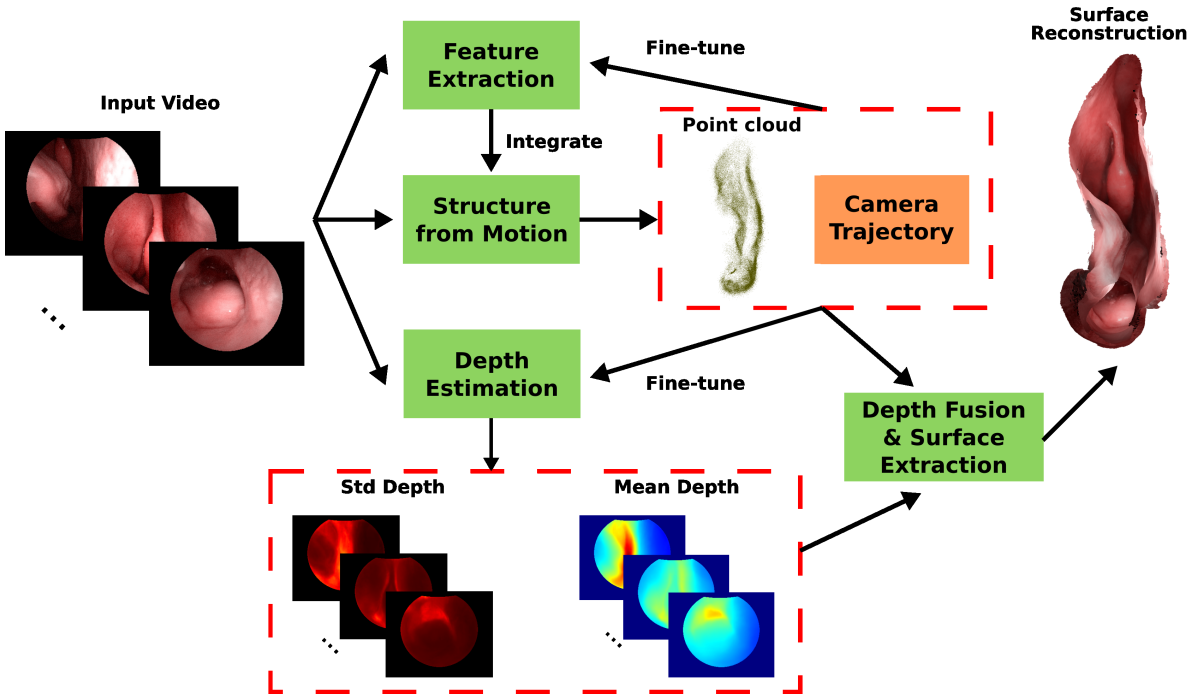

Figure 3.1: **Diagram of surface reconstruction generation.** Note that part of the surface reconstructions is removed in the figure to display internal structures.

if necessary until the performance is reasonable. The *Depth Estimation* module is then trained with the refined SfM estimate above to achieve good performance on the input video.

3.4 Self-supervised Monocular Depth Estimation with Uncertainty

In this section, we describe a method to train a neural network for monocular depth estimation in endoscopy with sparse supervisory signals derived from SfM estimates. We explain how supervisory signals from monocular endoscopy videos are extracted, and introduce our novel network architecture and loss functions to enable network training based on these signals.

3.4.1 Training Data

Our training data are generated from unlabeled endoscopic videos. The generation pipeline is shown in Fig. 3.2. The pipeline is fully automated given endoscopic and calibration videos and could, in principle, be computed on the fly with SLAM-based methods for sparse reconstruction.

Data Preprocessing. A video sequence is first undistorted using distortion coefficients estimated from the corresponding calibration video. A sparse reconstruction, camera trajectory, and the point visibility are estimated with the method described in Chapter 2 from a video. The video is undistorted before the processing and the regions beyond the mask boundary are ignored during sparse reconstruction. To re-

move extreme outliers in the sparse reconstruction, point cloud filtering is applied. The temporal point visibility information, which is labeled as b below, is smoothed out to exploit the fact that the camera movement is usually continuous in an endoscopic video. The sparse-form data generated from SfM results are introduced below.

Sparse Depth Map. Monocular depth estimation module, shown in Fig.3.3, only predicts depths up to a global scale. However, to enable valid loss calculation, the scale of the depth prediction and the SfM results must match. Therefore, the sparse depth map introduced here is used as an anchor to scale the depth prediction in the *Depth Scaling Layer*. To generate sparse depth maps, 3D points from the sparse reconstruction from SfM are projected onto image planes with camera poses, intrinsics, and point visibility information. The camera intrinsic matrix is $\boldsymbol{K}$; the camera pose of the world coordinate with respect to frame j is $\boldsymbol{T}^{j}_{\mathrm{wld}} \in \mathrm{SE}(3)$, where wld stands for world coordinate system; the coordinate of n^{th} 3D point of the sparse reconstruction in the world coordinate is $\boldsymbol{p}^{\mathrm{wld}}_{n} \in \mathbb{R}^3$. Note that n can be the index of any point in the sparse reconstruction. Frame indices used in the following equations, *e.g.*, j and k, can be any indices within the same video sequence. The difference of j and k is within a specified range to keep enough region overlap. The coordinate of n^{th} 3D point w.r.t.frame j, $\boldsymbol{p}^{j}_{n} \in \mathbb{R}^3$, is

$$\boldsymbol{p}^{j}_{n} = \boldsymbol{T}^{j}_{\mathrm{wld}} \boldsymbol{p}^{\mathrm{wld}}_{n} \quad . \tag{3.1}$$

The depth of n^{th} 3D point w.r.t.frame j, $z^{j}_{n} \in \mathbb{R}$, is the z-axis component of $\boldsymbol{p}^{j}_{n}$. The 2D

projection location of n^{th} 3D point w.r.t.frame j, $\boldsymbol{u}_n^j \in \mathbb{R}^2$, is

$$\boldsymbol{u}_n^j = \boldsymbol{K}\frac{\boldsymbol{p}_n^j}{z_n^j} \quad . \tag{3.2}$$

We use $b_n^j = 1$ to indicate that n^{th} 3D point is visible to frame j and $b_n^j = 0$ to indicate otherwise. Note that the point visibility information from SfM is used to assign the value to b_n^j. The sparse depth map of frame j, $\boldsymbol{Z}_j^{\text{s}} \in \mathbb{R}^{1\times H\times W}$, at 2D location $\boldsymbol{u}_n^j$ is

$$\boldsymbol{Z}_j^{\text{s}}\left(\boldsymbol{u}_n^j\right) = z_n^j \mathbb{1}\left[b_n^j = 1\right] \quad . \tag{3.3}$$

Note that Eq. 3.3, 3.4, and 3.5 describe the value assignments for regions where points of the sparse reconstruction project onto. For regions where no points project onto, the values are set to zero.

Sparse Flow Map. The sparse flow map is used in the Sparse Flow (SF) Loss introduced below. Previously, we directly used the sparse depth map for loss calculation [112] to exploit supervisory signals of sparse reconstructions. This makes the training objective, *i.e.*, sparse depth map, for one frame fixed and potentially biased. Unlike the sparse depth map, the sparse flow map describes the 2D projected movement of the sparse reconstruction, which involves camera poses of two input frames with a random frame interval. By combining the camera trajectory and sparse reconstruction, and considering all pair-wise frame combinations, the error distribution of

the new objective, *i.e.*, sparse flow map, for one frame is more likely to be unbiased. This makes the network less affected by the random noise in the training data. We observe that the depth predictions are naturally smooth with edge-preserving for the model trained with SF Loss, which removes the need for explicit regularization during training, such as the smoothness losses proposed in Zhou *et al*. [100] and Yin *et al*. [101].

The sparse flow map, $\boldsymbol{F}^{\mathrm{s}}_{j,k} \in \mathbb{R}^{2\times H\times W}$, represents the 2D projected movement of the sparse reconstruction from frame j to frame k.

$$\boldsymbol{F}^{\mathrm{s}}_{j,k}\left(\boldsymbol{u}^{j}_{n}\right) = \frac{\boldsymbol{u}^{k}_{n} - \boldsymbol{u}^{j}_{n}}{(W,H)^{\intercal}} \mathbb{1}\left[b^{j}_{n} = 1\right] \quad , \tag{3.4}$$

where H and W are the height and width of the frame, respectively.

Sparse Binary Mask. A sparse binary mask enables the network to exploit the valid sparse signals in the sparse-form data and ignore the rest of the invalid regions. The sparse mask of frame j, $\boldsymbol{M}_{j} \in \mathbb{R}^{1\times H\times W}$, is defined as

$$\boldsymbol{M}_{j}\left(\boldsymbol{u}^{j}_{n}\right) = b^{j}_{n} \quad . \tag{3.5}$$

3.4.2 Network Architecture

The overall network architecture, shown in Fig. 3.3, consists of a two-branch Siamese network [113] in the training phase. It relies on sparse signals from SfM and geometric constraints between two frames to learn to estimate depths from single endoscopic video frames. In the application phase, the network has a simple single-branch architecture for the depth estimation from a single frame. The depth estimate for each video frame consists of mean and standard deviation depth maps. This assumes the produced depth estimate follows a pixel-wise independent Gaussian distribution. All the custom layers below are differentiable so that the network can be trained in an end-to-end manner.

Monocular Depth Estimation. This module uses a modified version of the 57-layer architecture in [76], known as fully convolutional DenseNet, which achieves comparable performance with other popular architectures with a large reduction of network parameters by extensively reusing preceding feature maps. We replace the final activation, which was log-softmax, with linear activation to make the architecture suitable for the task of depth prediction. We change the number of channels in the last convolutional layer to 2 to produce the mean and standard deviation of the depth estimate. We also replace the transposed convolutional layers in the up transition part of the network with nearest neighbor upsampling and convolutional layers to reduce the checkerboard artifact of the output [114].

Depth Scaling Layer. This layer matches the scale of the depth prediction from *Monocular Depth Estimation* and the corresponding SfM results for correct loss calculation. Note that all operations of the following equations are element-wise except that $\sum$ here is the summation over all elements of a map. $\boldsymbol{Z}'_j \in \mathbb{R}^{1\times H\times W}$ is the depth prediction of frame j that is correct up to a scale. The scaled depth prediction of frame j, $\boldsymbol{Z}_j \in \mathbb{R}^{1\times H\times W}$, is

$$\boldsymbol{Z}_j = \left(\frac{1}{\sum \boldsymbol{M}_j}\sum\left(\boldsymbol{M}_j\frac{\boldsymbol{Z}^{\mathrm{s}}_j}{\boldsymbol{Z}'_j+\epsilon}\right)\right)\boldsymbol{Z}'_j \quad , \tag{3.6}$$

where $\epsilon \in \mathbb{R}$ is a small number to avoid zero division.

Flow from Depth Layer. To use the sparse flow map, generated from the SfM results, to guide network training with the SF Loss described later, the scaled depth map first needs to be converted to a dense flow map with the relative camera poses and the intrinsic matrix. This layer is similar to the one proposed in [101], where they use the produced dense flow map as the input to an optical flow estimation network. Here instead, we use it for the depth estimation training. The dense flow map is essentially a 2D displacement field describing a 3D viewpoint change. Given the scaled depth map of frame j, and the relative camera pose of frame k w.r.t.frame j, $\boldsymbol{T}^k_j = \left(\boldsymbol{R}^k_j, \boldsymbol{t}^k_j\right) \in \mathrm{SE}(3)$, a dense flow map between frame j and k, $\boldsymbol{F}_{j,k} \in \mathbb{R}^{2\times H\times W}$, can be derived. To demonstrate the operations in a parallelizable and differentiable way, the equations below are described in a matrix form.

The 2D locations in frame j, $(\boldsymbol{U}, \boldsymbol{V}) \in \mathbb{R}^{1\times H\times W}\times\mathbb{R}^{1\times H\times W}$, are organized as a regular

2D meshgrid. The corresponding 2D locations of frame k are $(\boldsymbol{U}_k, \boldsymbol{V}_k) \in \mathbb{R}^{1\times H\times W} \times \mathbb{R}^{1\times H\times W}$, which are organized in the same spatial arrangement as frame j. $\boldsymbol{U}_k$ and $\boldsymbol{V}_k$ are defined as

$$\begin{aligned} \boldsymbol{U}_k &= \frac{\boldsymbol{Z}_j \left(A_{0,0}\boldsymbol{U} + A_{0,1}\boldsymbol{V} + A_{0,2}\right) + B_{0,0}}{\boldsymbol{Z}_j \left(A_{2,0}\boldsymbol{U} + A_{2,1}\boldsymbol{V} + A_{2,2}\right) + B_{2,0}} \\ \boldsymbol{V}_k &= \frac{\boldsymbol{Z}_j \left(A_{1,0}\boldsymbol{U} + A_{1,1}\boldsymbol{V} + A_{1,2}\right) + B_{1,0}}{\boldsymbol{Z}_j \left(A_{2,0}\boldsymbol{U} + A_{2,1}\boldsymbol{V} + A_{2,2}\right) + B_{2,0}} \end{aligned} \quad . \tag{3.7}$$

As a regular meshgrid, $\boldsymbol{U}$ consists of H rows of $[0, 1, \ldots, W-1]$, and $\boldsymbol{V}$ consists of W columns of $[0, 1, \ldots, H-1]^\intercal$. $\boldsymbol{A} = \boldsymbol{K}\boldsymbol{R}_j^k\boldsymbol{K}^{-1} \in \mathbb{R}^{3\times 3}$ and $\boldsymbol{B} = -\boldsymbol{K}\boldsymbol{t}_j^k \in \mathbb{R}^3$. $A_{m,n} \in \mathbb{R}$ and $B_{m,n} \in \mathbb{R}$ are elements of A and B at position (m, n), respectively. The dense flow map, $\boldsymbol{F}_{j,k}$, for describing the 2D displacement field from frame j to frame k is

$$\boldsymbol{F}_{j,k} = \left(\frac{\boldsymbol{U}_k - \boldsymbol{U}}{W}, \frac{\boldsymbol{V}_k - \boldsymbol{V}}{H}\right) \quad . \tag{3.8}$$

Depth Warping Layer. The sparse flow map mainly provides guidance to regions of a frame where sparse information from SfM gets projected onto. Given that most frames only have a small percentage of pixels whose values are valid in a sparse flow map, most regions are still not properly guided. With the camera motion and camera intrinsics, geometric constraints between two frames can be exploited by enforcing consistency between the two corresponding depth predictions. The intuition is that the dense depth maps predicted separately from two neighboring frames are correlated because there is overlap between the observed regions. To make the geo-

metric constraints enforced in the Depth Consistency (DC) Loss described later differentiable, the viewpoints of the depth predictions must be aligned first.

Because a dense flow map describes a 2D projected movement of the observed 3D scene, $\boldsymbol{U}_k$ and $\boldsymbol{V}_k$ described above can be used to change the viewpoint of the depth $\boldsymbol{Z}_k$ from frame k to frame j with an additional step, which is modifying $\boldsymbol{Z}_k$ to describe the depth value changes due to the viewpoint changing. The modified depth map of frame k, $\tilde{\boldsymbol{Z}}_k \in \mathbb{R}^{1\times H\times W}$, is

$$\tilde{\boldsymbol{Z}}_k = \boldsymbol{Z}_k \left(C_{2,0}\boldsymbol{U} + C_{2,1}\boldsymbol{V} + C_{2,2}\right) + D_{2,0} \quad , \tag{3.9}$$

where $\boldsymbol{C} = \boldsymbol{K}\boldsymbol{R}_k^j\boldsymbol{K}^{-1} \in \mathbb{R}^{3\times 3}$, $\boldsymbol{D} = \boldsymbol{K}\boldsymbol{t}_k^j \in \mathbb{R}^3$. With $\boldsymbol{U}_k$, $\boldsymbol{V}_k$ and $\tilde{\boldsymbol{Z}}_k$, the bilinear sampler in [115] is able to generate the dense depth map $\check{\boldsymbol{Z}}_{k,j} \in \mathbb{R}^{1\times H\times W}$ that is warped from the viewpoint of frame k to that of frame j

3.4.3 Loss Design

We propose novel losses that exploit supervisory signals from SfM and enforce geometric consistency between depth predictions of two frames. A log-likelihood loss is used to enable the network to produce aleatoric uncertainty [116] of depth estimates.

Sparse Depth Loss. Sparse Depth (SD) Loss is used to encourage network producing estimates that agree with sparse reconstructions from SfM in terms of both

expectation and uncertainty. It is defined as

$$\mathcal{L}_{\mathrm{sd}}(j) = \frac{1}{\sum \boldsymbol{M}_j} \sum \left(\boldsymbol{M}_j \left(\ln\left(\boldsymbol{S}_j + \epsilon\right) + \frac{\left(\boldsymbol{Z}_j^{\mathrm{s}} - \boldsymbol{Z}_j\right)^2}{2\boldsymbol{S}_j^2 + \epsilon} \right) \right) \quad , \tag{3.10}$$

where $\boldsymbol{M}_j$, $\boldsymbol{Z}_j^{\mathrm{s}}$, $\boldsymbol{Z}_j$, and $\boldsymbol{S}_j \in \mathbb{R}^{1\times H\times W}$ are the sparse binary mask, sparse depth map, mean depth map, and the standard deviation depth map of frame j, respectively. ϵ is used to avoid numerical instability.

Sparse Flow Loss. To produce correct dense depth maps that agree with sparse reconstructions from SfM, the network is trained to minimize the differences between the dense flow maps and the corresponding sparse flow maps. This loss is scale-invariant because it considers the difference of the 2D projected movement in the unit of a pixel, which avoids the data imbalance problem caused by the arbitrary scales of SfM results. The SF Loss associated with frame j and k is calculated as

$$\begin{aligned} \mathcal{L}_{\mathrm{sf}}(j,k) = & \frac{1}{\sum \boldsymbol{M}_j} \sum \left(\boldsymbol{M}_j | \boldsymbol{F}_{j,k}^{\mathrm{s}} - \boldsymbol{F}_{j,k} | \right) + \\ & \frac{1}{\sum \boldsymbol{M}_k} \sum \left(\boldsymbol{M}_k | \boldsymbol{F}_{k,j}^{\mathrm{s}} - \boldsymbol{F}_{k,j} | \right) \quad . \end{aligned} \tag{3.11}$$

Depth Consistency Loss. Sparse signals from the SF Loss alone could not provide enough information to enable the network to reason about regions where no sparse annotations are available. Therefore, we enforce geometric constraints between two independently predicted depth maps. This loss, similar to SD Loss, is also

based on log-likelihood. The DC Loss associated with frame j and k is calculated as

$$\mathcal{L}_{\mathrm{dc}}(j,k) = \frac{1}{\sum \boldsymbol{W}_{k,j}} \sum \left(\boldsymbol{W}_{k,j} \left(\ln\left(\boldsymbol{S}_j + \epsilon\right) + \frac{\left(\check{\boldsymbol{Z}}_{k,j} - \boldsymbol{Z}_j\right)^2}{2\boldsymbol{S}_j^2 + \epsilon} \right) \right) \quad , \tag{3.12}$$

where $\boldsymbol{W}_{k,j} \in \mathbb{R}^{1\times H\times W}$ is the binary mask of the overlapping region of $\boldsymbol{Z}_j$ and the dense depth map $\check{\boldsymbol{Z}}_{k,j}$ that is predicted from frame k but warped to the viewpoint of frame j. Because SfM results are ambiguous in terms of global scaling, this loss only penalizes the relative difference between two dense depth maps to avoid data imbalance across training video sequences.

Overall Loss. The overall loss function for network training with a single pair of training data from frames j and k is

$$\mathcal{L}(j,k) = \lambda_1 \left(L_{\mathrm{sd}}(j) + L_{\mathrm{sd}}(k) \right) + \lambda_2 \mathcal{L}_{\mathrm{sf}}(j,k) + \lambda_3 \mathcal{L}_{\mathrm{dc}}(j,k) \quad . \tag{3.13}$$

3.5 Surface Reconstruction with Volumetric Truncated Signed Distance Field

3.5.1 Truncated Signed Distance Field

A distance field is an implicit surface representation. It is defined as a scalar field whose value at any given point is equal to the distance from the point to the nearest surface [117]. For surface extraction, a signed distance field is preferred because it avoids several drawbacks of an unsigned one: 1) Surface orientation is not preserved in the distance field without signs. 2) Recovering the most likely location of surfaces given a set of measurements affected by noise is not straightforward. The values of the surface locations in a signed distance field are always zero. And finding zero-crossing of a linear function is simpler than estimating the minimum of the piecewise linear function that results from the mean of absolute distances in the unsigned case. In the signed distance field, the surface orientation is also preserved, as it can be inferred from the positive direction of the gradient at the zero-crossing location.

Although the signed distance field is advantageous to an unsigned one, in practice, it is still difficult to construct such a field from partial observations of the environment. Curless and Levoy proposed a volumetric integration method for range images [92] that tackles this problem. They compute the line-of-sight distances within

the frustum of a sensor using surface measurements. Such a distance form is defined as a projective signed distance field, and this allows for local updates of the field based on partial observations. If the signed distance values in a field are truncated at small negative and positive values, it is then called Truncated Signed Distance Field. A partial observation, $\hat{\boldsymbol{D}}(\boldsymbol{x}) \in \mathbb{R}$, based on line-of-sight distances can be considered as a partial approximation of the truncated signed distance field (TSDF). When a new observation is obtained, weighted by a measurement weight, $\hat{\boldsymbol{W}}(\boldsymbol{x}) \in \mathbb{R}$, it will be added to the current estimate of TSDF, $\boldsymbol{D}_n(\boldsymbol{x}) \in \mathbb{R}$. The update rules for a given cell location $\boldsymbol{x} \in \mathbb{R}^3$ are defined as

$$\boldsymbol{D}_{n+1}(\boldsymbol{x}) = \frac{\boldsymbol{D}_n(\boldsymbol{x})\,\boldsymbol{W}_n(\boldsymbol{x}) + \hat{\boldsymbol{D}}(\boldsymbol{x})\,\hat{\boldsymbol{W}}(\boldsymbol{x})}{\boldsymbol{W}_n(\boldsymbol{x}) + \hat{W}(\boldsymbol{x})} \quad , \tag{3.14}$$

$$\boldsymbol{W}_{n+1}(\boldsymbol{x}) = \boldsymbol{W}_n(\boldsymbol{x}) + \hat{\boldsymbol{W}}(\boldsymbol{x}) \quad , \tag{3.15}$$

where $\boldsymbol{D}_{n+1}(\boldsymbol{x})$ is the updated TSDF at location $\boldsymbol{x}$ based on the projective estimate $\hat{\boldsymbol{D}}(\boldsymbol{x})$. The weight $\boldsymbol{W}_{n+1}(\boldsymbol{x})$ is the updated accumulated sum of measurement weights.

3.5.2 Volumetric Depth Fusion

We apply a depth fusion method based on TSDF [118] to build a volumetric representation of the surface model. Depth measurements are propagated to a 3D volume using ray-casting from the corresponding camera pose. The associated standard deviation depth is used as the slope of the truncated signed distance function for each ray. We used SfM results to re-scale all depth estimates before the fusion to make sure all estimates are scale-consistent. To fuse all information correctly, the camera poses estimated from SfM are used to propagate the corresponding depth estimates and color information to the 3D volume.

3.5.3 Explicit Surface Extraction

The Marching Cubes method [119] is used to extract a watertight triangular mesh surface from the TSDF volume computed above. This method is based on the premise that if one were to construct a cube using 8 neighboring voxels as vertices, there are only a few possible configurations for a surface passing through it. Classifying the inside and outside status of the neighboring vertices allows mapping the small region of the scalar field to a set of pre-determined configurations of triangular patches. The exact positions of the vertices that define these triangles can be adjusted to embed them into the zero level-set, by interpolation [117].

3.6 Experiments

We compared sparse reconstructions from SfM, dense reconstruction from COLMAP [19], and ground truth anatomy from CT, on *in* and *ex vivo* data, to demonstrate the effectiveness and accuracy of our method. We did not evaluate the monocular depth estimation method described here with separate experiments but instead evaluate the entire surface reconstruction pipeline as a whole. However, we did conduct a series of experiments for a previous version of the proposed depth estimation method presented in [16]. To briefly summarize here, in a cross-patient experiment using CT scans as groundtruth, the proposed method achieved submillimeter mean residual error. In a comparison study to recent self-supervised depth estimation methods [100, 101] designed for natural video on *in vivo* nasal endoscopy data, we demonstrate that the proposed depth estimation method outperforms the previous ones by a large margin. Please refer to Liu *et al*. [16] for details of the depth estimation experiments.

3.6.1 Experiment Setup

All experiments were conducted on one NVIDIA TITAN X GPU. The registration algorithm used for evaluation is based on Billings *et al*. [120], where a similarity transformation is optimized over. The endoscopic videos used in the experiments were

acquired from eight consenting patients and five cadavers under an IRB-approved protocol. The anatomy captured in the videos is the nasal cavity. The total time duration of videos is around 40 minutes. Because this method is patient-specific, all data are used for training. All processing related to the proposed pipeline used 4-time spatially downsampled videos, which have a resolution of 256×320.

SfM was first applied with SIFT features [121] to all videos to generate sparse reconstructions and camera trajectories. Results of this initial SfM run were used to train the descriptor network until convergence. The training setting for the descriptor network is the same as that in Chapter 2. For evaluation of each video sequence, SfM was applied again with the pre-trained descriptor network to generate a denser point cloud and a more complete camera trajectory. Note that if the trained descriptor network cannot produce satisfactory SfM results on the new sequence, descriptor network fine-tuning and another SfM run with the fine-tuned model are required, which was not required in our experiments.

The depth estimation network was then trained with the sequence-specific SfM result above. In terms of the hyperparameters of depth network training, the temporal range to smooth the point visibility information is set to 30; the frame interval of two frames that are randomly selected from the same sequence and fed to the two-branch training network is set to $[5, 30]$; the optimizer is SGD optimization with momentum set to 0.9 with cyclical learning rate [84] within the range of $[1.0\,\mathrm{e}^{-4}, 1.0\,\mathrm{e}^{-3}]$; The batch

size is set to 8; The ϵ in the depth scaling layer is set to 1.0e^{-8}; We train the network until the loss curve plateaus. λ_2 is set to 10.0. For the first 10 epochs, λ_1 and λ_3 are set to 0 to use SF Loss for the initial convergence. After that, λ_1 and λ_3 are set to 0.05 and 0.5, respectively, to start training the standard deviation depth branch and imposing dense geometric constraints between frames.

3.6.2 Comparison with Reconstruction from CT

Model accuracy was evaluated by comparing surface reconstructions with the corresponding CT models. In this evaluation, two metrics were used: average residual error between the registered surface reconstruction and the CT model, and the average relative difference between the corresponding cross-sectional areas of the CT surface models and the surface reconstructions. The purpose of this evaluation is to determine whether our reconstruction can be used as a low-cost, radiation-free replacement for CT when calculating clinically relevant parameters. To find the corresponding cross-section of two models, the surface reconstruction was first registered to the CT model. The registered camera poses from SfM were then used as the origins and orientations of the cross-sectional planes. The relative differences of all cross-sectional areas along the registered camera trajectory were averaged to obtain the final statistics.

This evaluation was conducted on 7 video sequences from 4 individuals. The resid-

ual error after registration was $0.69\,(\pm 0.14)$ mm. As a comparison, when the sparse reconstructions from SfM are directly registered to CT models, the residual error was $0.53\,(\pm 0.24)\,\mathrm{mm}$. The smaller error is due to the sparsity and smaller region coverage of the sparse reconstruction compared to ours. In Fig. 3.4, a visualization of the video-reconstruction-CT alignment is shown. The cross-sectional surface areas are estimated with an average relative error of $7\,(\pm 2)$ %. This error mainly originates from regions that were not sufficiently visualized during scoping, such as the inferior, middle, and superior meatus. These regions are included in our analysis due to the automation of cross-sectional measurements. In practice, these regions are not commonly inspected as they are hidden beneath the turbinates; if a precise measurement of these areas is desired, small modifications to video capture would allow for improved visualization. Similar to [110], such adjustments can be guided by our surface reconstruction, since the occupancy states in the fusion volume can indicate explicitly what regions were not yet captured with endoscopic video.

3.6.3 Comparison with Reconstruction from COLMAP

We used the ball pivoting [122] method to reconstruct surfaces in COLMAP instead of built-in Poisson [104] and Delaunay [123] methods because these two did not produce reasonable results. Three videos from 3 individuals were used in this evaluation. The qualitative comparison is shown in Fig. 3.5. The same scale recovery method

as above was used. The average residual distance after registration between the surface reconstructions from the proposed pipeline and COLMAP is $0.24\,(\pm 0.08)\,\mathrm{mm}$. In terms of the runtime performance, given that a pre-trained generalizable descriptor network and depth estimation network exist, our method requires running sparse SfM with a learning-based feature descriptor, fine-tuning depth estimation network, depth fusion, and surface extraction. For the three sequences, the average runtime for the proposed method is 127 minutes, whereas the runtime for COLMAP is 778 minutes.

3.6.4 Reconstruction Consistency Against Video Variation

Surface reconstruction methods should be insensitive to variations in video capture, such as camera speed. To evaluate the sensitivity of our method, we randomly sub-sampled frames from the original video to mimic camera speed variation. The pipeline was run for each sub-sampled video and we evaluated the model consistency by aligning surface reconstructions estimated from different subsets. To simulate camera speed variation, out of every 10 consecutive video frames, only 7 frames were randomly selected. We evaluated the model consistency on 3 video sequences that cover the entire nasal cavity of three individuals, respectively. Five reconstructions

that were computed from random subsets of each video were used for evaluation. The average residual distance after registration between different surface reconstructions was used as the metric for consistency. The scale recovery method is the same as above. The residual error was $0.21\,(\pm 0.10)$ mm.

3.6.5 Agreement Between Surface Reconstruction and Supervisory Signal

Because our method is self-supervised and patient-specific, and SfM results are used to derive supervisory signals, the discrepancy between the surface and sparse SfM reconstruction should be minimal. To evaluate the consistency between our surface reconstruction and the sparse reconstruction, we calculated the point-to-mesh distance between the two. Because scale ambiguity is intrinsic for monocular-based surface reconstruction methods, we used the CT surface models to recover the actual scale for all individuals where CT data are available. For those that do not have corresponding CT data, we used the average statistics of the population to recover the scale. The evaluation was conducted on 33 videos of 13 individuals. The estimated point-to-mesh distance was $0.34\,(\pm 0.14)$ mm. Examples of the sparse and surface reconstruction overlaid with point-to-mesh distance are shown in Fig. 3.6.

3.7 Discussion

3.7.1 Choice of Depth Estimation Method

In this work, a monocular depth estimation network is used to learn the complex mapping between the color appearance of a video frame and the corresponding dense depth map. The method in [16] has been shown to generalize well to unseen cases. However, the patient-specific training in this pipeline may allow for higher variance mappings since it does not need to generalize to other unseen cases. Therefore, a more complex network architecture could potentially further improve the depth estimation accuracy, leading to more accurate surface reconstruction. For example, a self-supervised recurrent neural network that predicts the dense depth map of a video frame based on the current observation and the previous frames in the video could potentially have more expressivity and be able to learn a more complex mapping, such as the method proposed by Wang *et al*. [124].

3.7.2 Limitations

The proposed pipeline will fail if SfM cannot generate reasonable sparse reconstruction and camera trajectory. This could happen in some cases, such as fast camera movements and blurry images, which are already mitigated with the proposed

dense descriptor being used. Large tissue deformation during video capturing could also make SfM fail, a non-rigid SfM algorithm can be used to mitigate this issue. The pipeline currently only uses depth uncertainty estimates to make better depth merging in TSDF and the reconstructed surface model does not contain surface uncertainty information. A volumetric surface uncertainty estimation method could be developed for this purpose.

For evaluation, the residual error used as the metric for CT evaluation can lead to underestimated errors. This is because the residual error is calculated using pairs of closest points between the registered point clouds and the CT surface models. Since the distance between the closest point pair is always less than or equal to the distance between the true point pair, the overall error will be underestimated. The exact accuracy estimate is available only if the camera trajectory of a video is accurately registered to the CT surface model, which is what we currently do not have.

3.8 Conclusion

In this chapter, we develop a method for reconstructing an accurate surface model from a monocular endoscopic video. To our knowledge, we propose the first self-supervised monocular depth estimation method for endoscopy. Our method operates directly on raw endoscopic videos and produces watertight textured surface models

that are in good agreement with anatomy extracted from CT. While this work so far has only been evaluated on videos of the nasal cavity, the proposed pipeline is generic and should thus apply to other anatomies as well.

The surface reconstruction pipeline currently only works in a static environment because of the depth fusion method [92] being used. It would be interesting to explore whether an optimizable dense depth representation, used in Chapter 5, can be integrated into the non-linear optimization of the SfM to treat the depth code as an additional type of variable to optimize for. If there are non-deformation changes, such as bleeding and instrument moving during video capturing, whether the pipeline can work reasonably well depends on how large the affected regions are. The pipeline will certainly fail if such changes lead to systematic erroneous sparse reconstruction results from SfM. However, if only a small percentage of extreme outliers exist in the SfM results, it is expected that the depth estimation training scheme can tolerate such training noise and be affected minimally. If the camera trajectory estimate from SfM is also decently accurate, the pipeline should still produce high-quality surface reconstruction with blurry textures only for these changing regions. The current procedures of depth estimation and depth fusion are disjoint and no shape prior of the anatomy to be reconstructed is involved. It could be interesting to explore how to effectively integrate a deep shape prior of the anatomy into the pipeline to better respect the topology and more effectively handle the errors in the depth estimates.

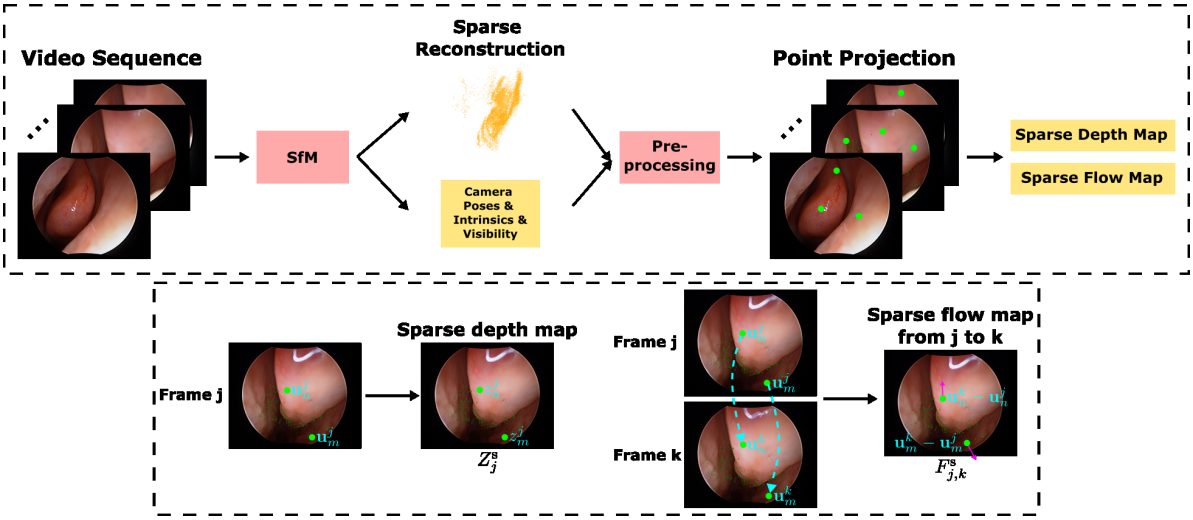

Figure 3.2: **Pipeline of training data generation for depth estimation.** The pipeline can generate training data from video sequences automatically. The symbols in the figure are defined in the Sec. 3.4.1. The green dots are shown in the figure stand for example projected 2D locations of the sparse reconstruction. These projected 2D locations are used to store valid information for all the sparse-form data, *i.e.*, sparse depth map, sparse binary mask, and sparse flow map. A sparse depth map stores z-axis distances of the sparse reconstruction w.r.t.the camera coordinate. A sparse flow map stores movement of projection locations of the sparse reconstruction between two frames. A sparse binary mask stores binary weights which indicate the projective locations of individual points in the sparse reconstruction, which is not shown in this figure. The generation of a sparse depth map and sparse flow map is shown in the second row of the figure, where projected location samples are used to demonstrate the concept. The cyan dash arrows are used to indicate point correspondences between two frames. Note that the sparse-form data do not include the color video frames in the figure, which is used only for display purposes. © 2019 IEEE

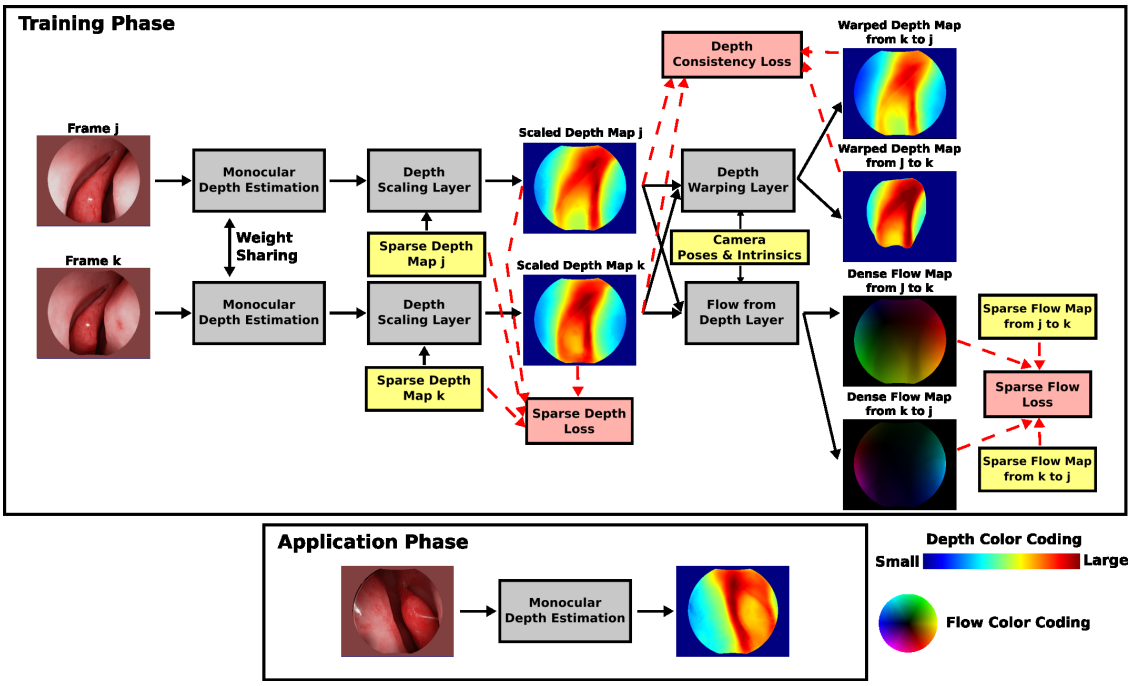

Figure 3.3: **Diagram for training and applying depth estimation.** Our network in the training phase (top) is a self-supervised two-branch Siamese network. Two frames j and k are randomly selected from the same video sequence as the input to the two-branch network. To ensure enough region overlap between two frames, the frame interval is within a specified range. All concepts in the figure are introduced in Sec. 3.4. The red dashed arrows are used to indicate the data-loss correspondence. The warped depth map from k to j describes the scaled depth map k viewed from the viewpoint of frame j. The dense flow map from j to k describes the 2D projection movement of the underlying 3D scene from frame j to k. During the application phase (bottom), we use the trained weights of the single-frame depth estimation architecture, which is a modified version of the architecture in Jégou *et al*. [76], to predict mean and standard deviation depth maps that are correct up to a global scale. Note that standard deviation depth maps are not displayed in the figure.

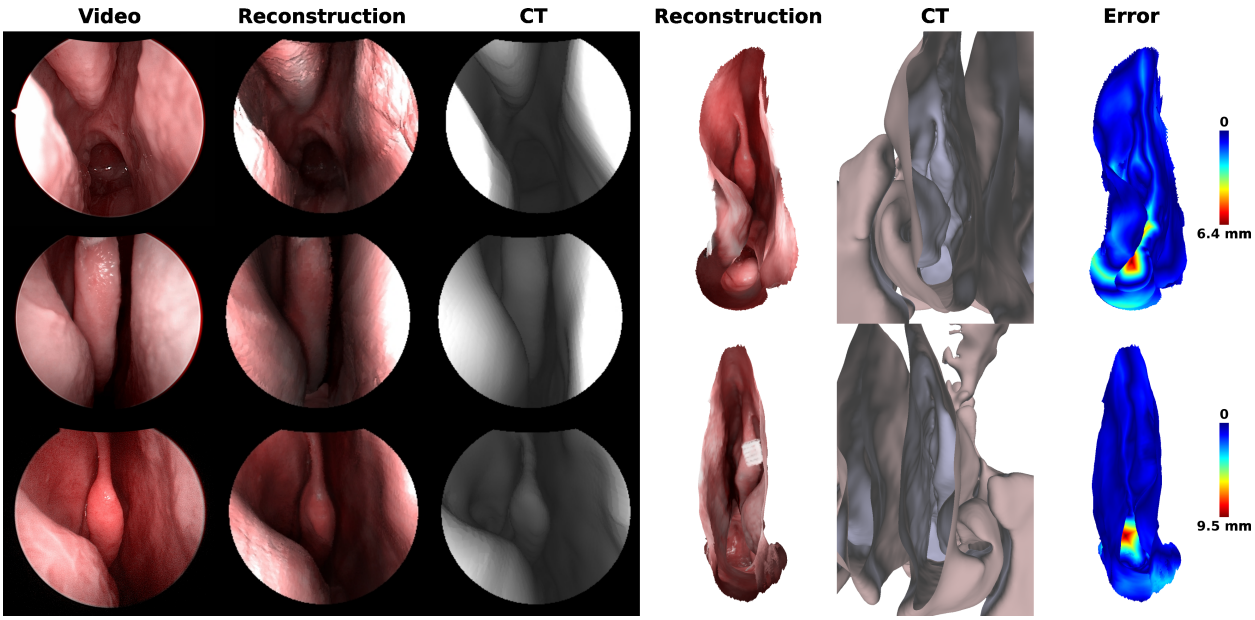

Figure 3.4: **Visualization of registered frames, surface reconstructions, CT models, and residual errors.** To produce such visualization, the dense reconstruction is registered to the CT model to obtain the transform between the two models. Afterward, for the first three columns in the figure, the reconstruction and camera trajectory estimate are aligned with the CT model, and the side-by-side display in the figure can be generated. To provide more context information in terms of alignment, in the last three columns, we display the surface reconstruction, CT model, and the reconstruction with residual error overlay from a distant viewpoint. The residual error represents the mesh-to-mesh distance between the surface reconstruction and the CT model.

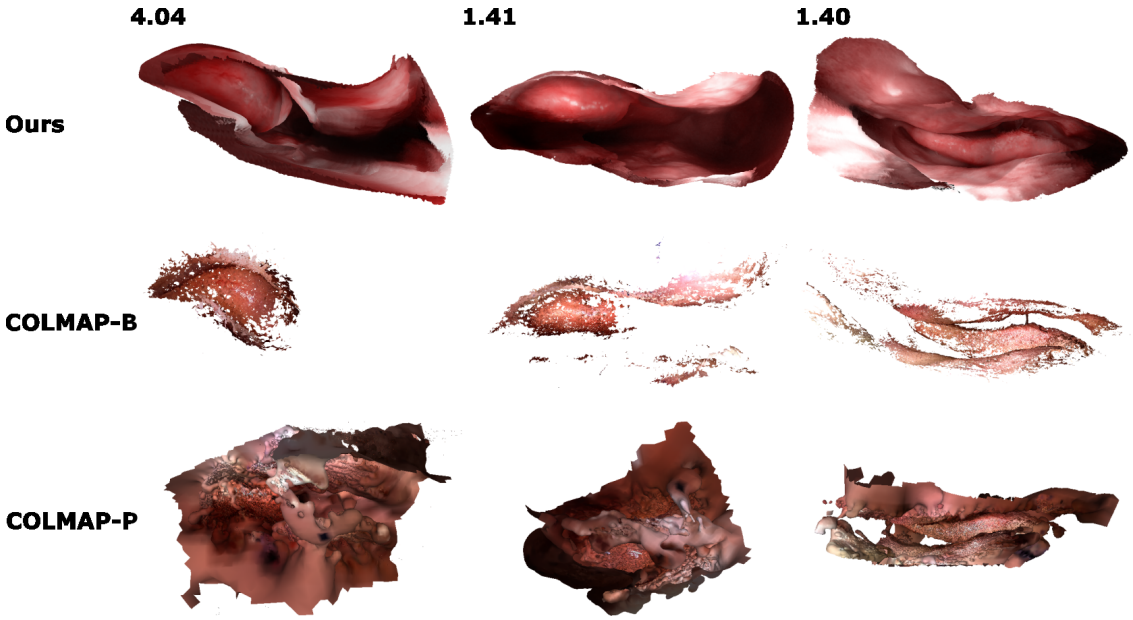

Figure 3.5: **Comparison of surface reconstructions from evaluated methods.** The number in each column is the ratio of surface area between our reconstruction and COLMAP with ball pivoting [122] (COLMAP-B). Ratios are underestimated because many redundant invalid surfaces are generated in the second row. COLMAP with Poisson [104] (COLMAP-P) is shown in the last row with excessive surfaces removed already.

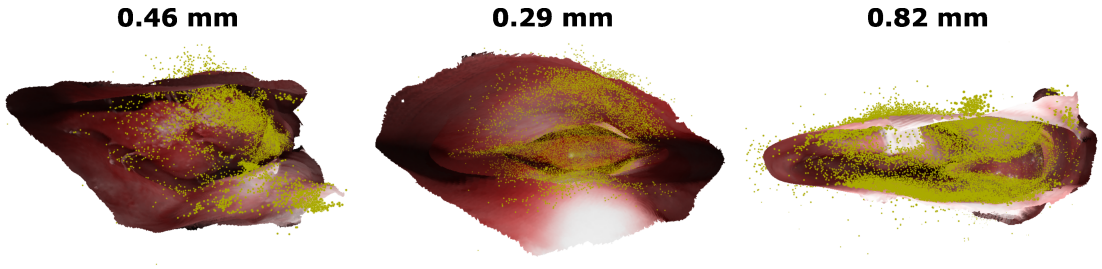

Figure 3.6: **Overlay of sparse reconstruction and surface reconstruction.** Sparse reconstruction from SfM is overlaid with surface reconstruction from the proposed pipeline. The number in each column represents the average point-to-mesh distance.

Chapter 4

Global 3D Registration with Deep Geometric Features

The surface reconstruction method developed in Chapter 3 enables a surface model to be computed from a monocular endoscopic video. Besides endoscopic video, other imaging modalities, such as CT, are often jointly used for disease diagnosis, surgical planning, *etc*. Having information from different modalities aligned and fused is beneficial in many aspects. For example, if a CT volume is aligned with the surface reconstruction from a video, the information of structures beneath the tissue surface will be known. This can help avoid critical structures if the alignment is conducted during endoscopy or can be used for retrospective analysis if obtained offline. Con-

ventionally, surgeons often go through a mental process to spatially align different modalities so that the correspondences are known. In quantitative endoscopy, such a process should be handled automatically by the computer.

Multi-modal registration [125] is the field where information from multiple modalities are aligned spatially. Point cloud registration algorithms have been studied for decades, however, most methods are developed for data from the same type of modality. And the multi-modal point cloud registration receives less attention [125], especially in the branch of learning-based methods. Compared with the same-modal method, the multi-modal one has additional challenges, such as different point cloud densities and patterns of noise and outliers.

In this chapter, we aim at the task of global point cloud registration from multiple modalities, specifically video-CT registration where the reconstruction from a monocular video is aligned to the surface model from CT data. The main challenge with video-CT registration is the unrecognized resolution mismatch of samples because of the scale ambiguity of reconstructions from monocular videos. To confront this challenge, we developed a deep geometric feature descriptor with a novel network normalization technique. Such deep geometric features enable robust correspondence estimation across point clouds with resolution mismatch, based on our evaluation. When such features are integrated into a global point cloud registration algorithm that requires point correspondences, a robust global point cloud registration will then

become available.

4.1 Related Work

4.1.1 Geometric Feature Descriptor

A great deal of research in 3D descriptors focuses on hand-crafted geometric descriptors. Local hand-crafted descriptors often process low-level geometric features, such as location, normal orientation, and curvature, with a hand-crafted algorithmic pipeline. In general, these types of descriptors are robust to partial correspondence but have relatively low distinguishability. Other hand-crafted methods use a global representation of shape, such as a functional map [126–131], to generate dense correspondences between shapes that may undergo isometric or non-isometric deformation. These methods have better descriptiveness but are often considered unsuitable in presence of partial correspondence.

Previous approaches to learning-based descriptors use data-driven parameterization to enhance the performance of hand-crafted descriptors [132–135]. In recent years, learning-based methods gained wider popularity with the advent of deep learning, which facilitated both local and global descriptors. Local descriptors [136–141] focus on extracting feature descriptors from a local patch around the query point and

usually have high generalizability across datasets. However, because of the small context information used, they are not suitable for producing consistent descriptors for samples with resolution mismatch.

Global descriptors [59], on the other hand, aim to process the entire 3D data with a neural network in one forward pass, producing element-wise dense feature descriptions. In this branch, many works aim at shape correspondence. Research has explored global 3D architectures for learning deep functional maps which can estimate dense correspondences on shape pairs under various deformations [142–147]. Some of these works [143, 146] combine deep learning with functional maps and are reasonably robust to partial correspondence, but the requirement of mesh connectivity information renders them inapplicable to point cloud registration. FCGF [59], which uses a global sparse voxelized architecture, aims at the task of sparse correspondence estimation in point clouds, the context of our work, and demonstrated superior performance on recent point cloud registration benchmarks [136, 148].

4.1.2 Network Normalization

Many global normalization techniques have been developed for neural networks. Batch Normalization (BN) [149] uses mini-batch statistics to approximate the larger distribution during training. It has been shown to help reduce the internal covariate shift and smooth out the loss landscape [150], easing the optimization problem. Layer

Normalization (LN) [151] normalizes along all dimensions of a sample. Instance Normalization (IN) [152] was originally developed for style transfer. It is similar to layer normalization but, instead of normalizing each sample, normalizes each channel in a sample independently. Group Normalization (GN) [153] normalizes channels as different groups within a sample, tending to perform better than batch normalization in the case of small batch size.

Local normalization techniques have also been proposed, such as Local Response Normalization (LRN) [154] and Local Context Normalization (LCN) [155], where the value of the center pixel or voxel is normalized using the statistics of its neighborhoods along either channel dimension, spatial dimension, or both.

Fusion methods sidestep the issue of normalization selection by combining multiple techniques in a learnable proportion, potentially letting the network use the advantages of each. These fusion techniques include Batch-Instance Normalization (BIN) [156], Switchable Normalization (SN) [157], and Sparse Switchable Normalization (SSN) [158], and have shown better performance compared to using a single type of normalization in certain tasks.

4.1.3 3D Network Architecture

The choice of network architecture is important because it greatly affects the ability to learn informative representations of data. Many 3D architectures have been

proposed. These can be approximately grouped into four branches based on the input data format: 3D volume [159–163], raw point cloud [164–169], mesh [144, 170–172], and graph [173–177]. These can be approximately grouped into three branches based on the input data format: 3D volume, raw point cloud, and mesh.

Perhaps the most straightforward way to process 3D data is to transform it into a voxel grid representation and then apply a dense 3D Convolutional Neural Network (CNN) [159–161]. However, this is expensive in terms of both memory and computation. Recent advances in the voxel format focus on sparse representations of the 3D volume [162, 163]. This allows for a 3D volume with a much finer resolution to be defined without exhausting computational resources.

Another approach that remains computationally efficient involves direct computation on the point cloud data. Since the pioneering work of PointNet [164], several architectures have emerged that operate in this domain [165–169]. Broadly, these involve a permutation-invariant aggregation or convolution over local neighborhoods to effectively represent the geometric structure of points.

Finally, the mesh surface representation provides connectivity information in addition to point location. CNN-based methods in this area [144, 170–172] apply convolution operations on the vertex or edge neighbors that are stored in the mesh data format. Graph-based methods [173–177] treat vertices as graph nodes and vertex connections as graph edges, leveraging graph neural networks rather than traditional

CNNs.

4.1.4 Point Cloud Registration

Point cloud registration methods can be roughly grouped into three types: 1) optimization-based, 2) end-to-end learning-based, and 3) hybrid registration. Optimization-based methods treat the registration problem as an objective that can be minimized with an optimization solver. Under this category, four subtypes can be defined: Iterative Closest Point (ICP)-based [178–182], graph-based [183–186], Gaussian Mixture Model (GMM)-based [187–190], and semi-definite based registration [191–198].

End-to-end learning-based methods [199–201] often feed two point clouds as a single input to the neural network, and a transformation will be directly predicted from it. This converts the registration problem to a regression task.

Hybrid methods [59, 136, 138–140, 202–208] often combine the representation power of a neural network with an conventional optimization process. Some methods first extract distinctive features from point clouds and then estimate the transformation with an outlier-robust estimation method such as RANSAC [66]. Volumetric representation and point cloud without voxelization are two typical choices. There are also methods proposing to produce other representations, such as soft correspondence assignments [209, 210], global features [211, 212], and optimization priors [213] *etc*.,

from the network and continue with an optimization process to complete the registration. Our method belongs to the hybrid registration group because we rely on geometric features estimated by a neural network to establish point-wise correspondences.

4.2 Contributions

Our main contributions are as follows:

- We present a novel normalization technique, Batch-Neighborhood Normalization (B-NHN), that shows to increase the network's robustness to task-irrelevant mean-std variation of local feature distribution, where resolution mismatch that we try to deal with is a specific data variation potentially causing that. This technique is general and can be applied to any neural network architecture having the concept of convolution over local neighbors. Specifically, on 3DMatch [136] and KITTI odometry [148] datasets for geometric descriptor benchmarking, we show that our method performs favorably against the state-of-the-art on the standard benchmarks, and outperforms previous methods by a large margin on the created resolution-mismatch ones.

- We contribute a dataset of nasal cavities built from CT scans to benchmark the performance of geometric feature extraction methods on a medical video-CT

registration task, where resolution mismatch is common.

From the system point of view, this work enables the automatic alignment of 3D models. Specifically, for Video-CT registration, the sparse or surface reconstruction estimated from a video can be automatically registered to the surface model obtained from a CT scan. This improves the awareness of surgeons on the surrounding anatomical structures underneath the visible surface during endoscopy procedures. Moreover, this work potentially allows for non-rigid registration of video reconstruction to the statistical shape model built from CT data from a large population and removes the need for a patient-specific CT scan.

4.3 Robustness to Resolution Variation Through Normalization

Resolution mismatch, *i.e.*, point density mismatch in the point cloud, is a specific type of data variation where the number of points per unit volume varies. When the global scales of samples are known, a simple convolution alternate, *e.g.*, averaging aggregation mode in a PointConv-based architecture, may already be good enough. However, in this work, we focus on the more challenging case where the scales are unknown and thus the resolution mismatch cannot be even recognized.

In this case, the convolutional neural network may need to learn many sets of filters to produce consistent feature descriptions across samples. Because the same actual receptive field may only be covered in different network layers for samples with resolution mismatch, the intermediate feature representations in the same layer for these samples will vary. This leads to the potential variation of the mean and standard deviation of features.

Therefore, we argue that removing or reducing such variation before any convolution operation may reduce the potential task-irrelevant information and thus improve robustness. It could also help the network distribute more resources on filters focusing on other variations that cannot be handled with normalization, such as spatial orientation. As a result, addressing variation introduced by resolution mismatch directly via normalization will potentially increase its capability of producing consistent features for samples with varying resolution and improve the expressivity of the network.

Based on the concepts above, we propose a new type of local normalization technique, (Batch-)Neighborhood Normalization ((B-)NHN). Unlike the previous local normalization techniques, such as LRN [154] and LCN [155], that treat the normalization as a standalone module, (B-)NHN is tightly coupled with the subsequent convolution operation, as shown in Fig. 4.1. Before a convolution kernel is applied to a volume patch, (B-)NHN is first applied to ensure the features in the patch are normalized. In

the rest of this section, we describe NHN and B-NHN in the context of sparse voxelized 3D convolution. We refer to NHN and B-NHN as the basis for the NHN-Conv and B-NHN-Conv modules, respectively. In our experiments, we mainly demonstrate the benefit of the proposed normalization in the resolution-mismatch setting, but we emphasize that the technique is general and should also improve the robustness of a network to other types of data variation that affect the mean and standard deviation of local regions.

4.3.1 Neighborhood Normalization

Let $\boldsymbol{x}_{\boldsymbol{u}}^{\text{in}} \in \mathbb{R}^{C_{\text{in}}}$ be a C_{in}-dimensional feature vector at 3D location $\boldsymbol{u} \in \mathbb{R}^3$. Denote the convolution kernel weights as $\boldsymbol{W} \in \mathbb{R}^{M \times C_{\text{out}} \times C_{\text{in}}}$, where M is the size of the local neighborhood, and let $\boldsymbol{W_i} \in \mathbb{R}^{C_{\text{out}} \times C_{\text{in}}}$ denote the kernel weights at spatial offset $\boldsymbol{i}$ from center. Thus, the output of a regular voxelized sparse 3D convolution at $\boldsymbol{u}$ is:

$$\boldsymbol{x}_{\boldsymbol{u}}^{\text{out}} = \sum_{\boldsymbol{v} \in \mathcal{N}(\boldsymbol{u})} \boldsymbol{W}_{\boldsymbol{v}-\boldsymbol{u}} \boldsymbol{x}_{\boldsymbol{v}}^{\text{in}} \quad , \tag{4.1}$$

where $\mathcal{N}(\boldsymbol{u})$ is the local neighborhood of the voxel at $\boldsymbol{u}$. Let $\mu_{\mathcal{N}(\boldsymbol{u})} \in \mathbb{R}$ and $\sigma_{\mathcal{N}(\boldsymbol{u})} \in \mathbb{R}$ be the mean and standard deviation of the neighborhood, $\mathcal{N}(\boldsymbol{u})$, over both channel

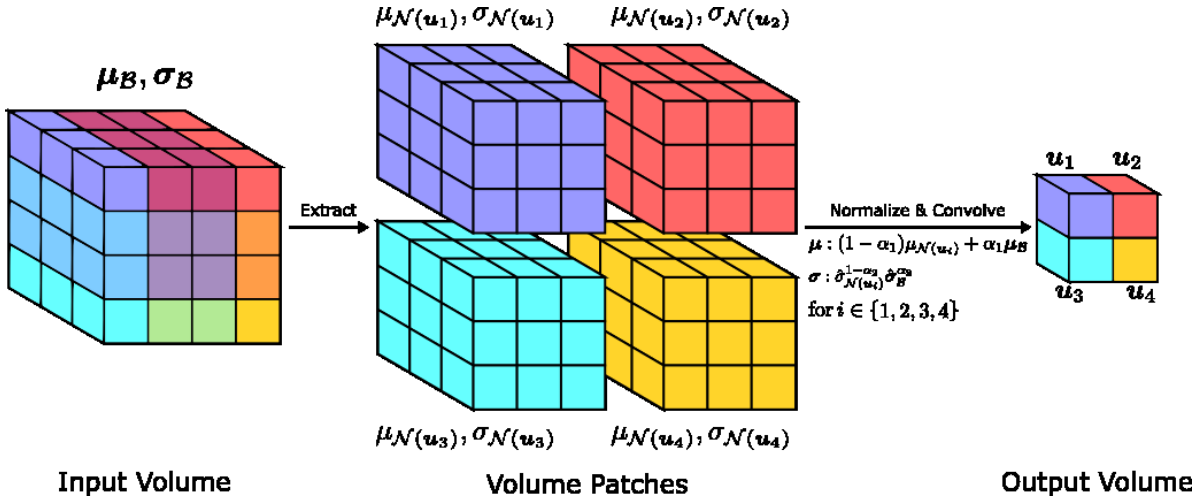

Figure 4.1: **Procedure of Batch-Neighborhood Normalization with convolution.** We use a dense 3D volume, with the size of 3x4x4, here to demonstrate the procedure handled naively, as in Eq. 4.4 and 4.6. The indicated 3D convolution here has a kernel size of 3x3x3 with no padding, and with both stride and dilation as 1. The input and output channel information is not shown here for simplicity. In this figure, the input volume first computes its channel-wise batch-norm statistics, $\boldsymbol{\mu}_{\mathcal{B}}, \boldsymbol{\sigma}_{\mathcal{B}} \in \mathbb{R}^{C_{\text{in}}}$. Four 3x3x3 volume patches are then extracted from the input volume with voxel overlap. The individual local statistics, $\mu_{\mathcal{N}(\boldsymbol{u}_i)}, \sigma_{\mathcal{N}(\boldsymbol{u}_i)} \in \mathbb{R}$, for each volume patch is then computed. After fusing the mentioned two types of statistics with learnable parameters $\alpha_1, \alpha_2 \in \mathbb{R}$, the volume patches are normalized with the corresponding fused statistics and then convolved with the 3D convolution filters to produce the output volume, which has a spatial size of 1x2x2. NHN is a special case of B-NHN where α_1, α_2 are all zeros. Please refer to Sec. 4.3.1 for the definitions of the symbols above.

and spatial dimensions.

$$\mu_{\mathcal{N}(\boldsymbol{u})} = \frac{1}{|\mathcal{N}(\boldsymbol{u})|C_{\text{in}}} \sum_{\boldsymbol{v} \in \mathcal{N}(\boldsymbol{u})} \mathbf{1}^{\intercal} \boldsymbol{x}_{\boldsymbol{v}}^{\text{in}} \tag{4.2}$$

$$\sigma^2_{\mathcal{N}(\boldsymbol{u})} = \frac{1}{|\mathcal{N}(\boldsymbol{u})|C_{\text{in}}} \sum_{\boldsymbol{v}\in\mathcal{N}(\boldsymbol{u})} \mathbf{1}^\intercal \left(\boldsymbol{x}^{\text{in}}_{\boldsymbol{v}} - \mu_{\mathcal{N}(\boldsymbol{u})}\mathbf{1}\right)^2 \tag{4.3}$$

where $\mathbf{1} \in \mathbb{R}^{C_{\text{in}}}$ is the all-ones vector. The NHN-Conv operation, described below, consists of neighborhood normalization followed by a convolution.

$$\boldsymbol{x}^{\text{out}}_{\boldsymbol{u}} = \sum_{\boldsymbol{v}\in\mathcal{N}(\boldsymbol{u})} \boldsymbol{W}_{\boldsymbol{v}-\boldsymbol{u}} \left(\boldsymbol{\gamma} \circ \left(\frac{\boldsymbol{x}^{\text{in}}_{\boldsymbol{v}} - \mu_{\mathcal{N}(\boldsymbol{u})}}{\hat{\sigma}_{\mathcal{N}(\boldsymbol{u})}}\right) + \boldsymbol{\beta}\right) \quad , \tag{4.4}$$

where $\boldsymbol{\gamma} \in \mathbb{R}^{C_{\text{in}}}$ and $\boldsymbol{\beta} \in \mathbb{R}^{C_{\text{in}}}$ are per-channel scaling and bias weights, $\hat{\sigma}_{\mathcal{N}(\boldsymbol{u})} = \sqrt{\sigma^2_{\mathcal{N}(\boldsymbol{u})} + \epsilon}$, and ϵ prevents zero division. $\circ$ denotes the Hadamard product, or element-wise multiplication. Although Eq. 4.4 is intuitively simple, it requires inefficient duplication of memory, since the same input voxel may need to be normalized with different statistics for each overlapping neighborhood window. Fortunately, a reformulation of the equation solves this issue.

$$\begin{aligned}\boldsymbol{x}^{\text{out}}_{\boldsymbol{u}} = \frac{1}{\hat{\sigma}_{\mathcal{N}(\boldsymbol{u})}} \sum_{\boldsymbol{v}\in\mathcal{N}(\boldsymbol{u})} \boldsymbol{W}_{\boldsymbol{v}-\boldsymbol{u}} \left(\boldsymbol{\gamma} \circ \boldsymbol{x}^{\text{in}}_{\boldsymbol{v}}\right) \\ - \frac{\mu_{\mathcal{N}(\boldsymbol{u})}}{\hat{\sigma}_{\mathcal{N}(\boldsymbol{u})}} \sum_{\boldsymbol{v}\in\mathcal{N}(\boldsymbol{u})} \boldsymbol{W}_{\boldsymbol{v}-\boldsymbol{u}}\boldsymbol{\gamma} + \sum_{\boldsymbol{v}\in\mathcal{N}(\boldsymbol{u})} \boldsymbol{W}_{\boldsymbol{v}-\boldsymbol{u}}\boldsymbol{\beta} \quad .\end{aligned} \tag{4.5}$$

The equivalent formulation in Eq. 4.5 is desirable because it decouples the computation of local mean and standard deviation, $\mu_{\mathcal{N}(\boldsymbol{u})}$, and $\sigma_{\mathcal{N}(\boldsymbol{u})}$, from the 3D convolution operation, removing unnecessary duplication. Further implementation details are

provided in Appendix A.

4.3.2 Batch-Neighborhood Normalization

Though NHN is well-suited to developing resolution-robust features, it has the inherent property of removing local mean and variance information before applying convolution. In some applications, it may be beneficial to preserve a portion of that information, which leads to the introduction of Batch-Neighborhood Normalization (B-NHN). As one might expect, B-NHN fuses the channel-wise batch-norm statistics [149] and the sample-wise statistics of the local neighborhood in a learnable manner. Like NHN, B-NHN is not a standalone operation but is performed along with the subsequent convolution, resulting in the B-NHN-Conv layer:

$$\boldsymbol{x}_{\boldsymbol{u}}^{\text{out}} = \sum_{\boldsymbol{v} \in \mathcal{N}(\boldsymbol{u})} \boldsymbol{W}_{\boldsymbol{v}-\boldsymbol{u}} \hat{\boldsymbol{x}}_{\boldsymbol{v}}^{\text{in}} \quad \text{, where} \tag{4.6}$$

$$\hat{\boldsymbol{x}}_{\boldsymbol{v}}^{\text{in}} = \boldsymbol{\gamma} \left(\frac{\boldsymbol{x}_{\boldsymbol{v}}^{\text{in}} - (1-\alpha_1)\mu_{\mathcal{N}(\boldsymbol{u})} - \alpha_1 \boldsymbol{\mu}_{\mathcal{B}}}{\hat{\sigma}_{\mathcal{N}(\boldsymbol{u})}{}^{1-\alpha_2} \cdot \hat{\boldsymbol{\sigma}}_{\mathcal{B}}^{\alpha_2}} \right) + \boldsymbol{\beta} \quad , \tag{4.7}$$

where $\boldsymbol{\mu}_{\mathcal{B}} \in \mathbb{R}^{C_{\text{in}}}$ and $\boldsymbol{\sigma}_{\mathcal{B}} \in \mathbb{R}^{C_{\text{in}}}$ are the batch-norm statistics, and $\hat{\boldsymbol{\sigma}}_{\mathcal{B}} = \sqrt{\boldsymbol{\sigma}_{\mathcal{B}}^2 + \epsilon}$. $\alpha_1 \in \mathbb{R}$ and $\alpha_2 \in \mathbb{R}$ are the learnable fusion parameters that control the portion of batch and neighborhood information in the mean and standard deviation, respectively. Note the vector division is simple element-wise or Hadamard division. Fig. 4.1 visualizes the B-NHN-Conv operation for multiple neighborhoods in a single-channel input volume.

α_1 controls an arithmetic weighting of batch and neighborhood means, and α_2 controls a geometric mean of the standard deviations. The geometric mean here is to enable an efficient reformulation, similar to Eq. 4.5:

$$\boldsymbol{x}_{\boldsymbol{u}}^{\text{out}} = \frac{1}{\hat{\sigma}_{\mathcal{N}(\boldsymbol{u})}^{1-\alpha_2}} \sum_{\boldsymbol{v} \in \mathcal{N}(\boldsymbol{u})} \boldsymbol{W}_{\boldsymbol{v}-\boldsymbol{u}} \left(\frac{\boldsymbol{\gamma}}{\hat{\boldsymbol{\sigma}}_{\mathcal{B}}^{\alpha_2}} \left(\boldsymbol{x}_{\boldsymbol{v}}^{\text{in}} - \alpha_1 \boldsymbol{\mu}_{\mathcal{B}} \right) + \boldsymbol{\beta} \right) - (1-\alpha_1) \frac{\mu_{\mathcal{N}(\boldsymbol{u})}}{\hat{\sigma}_{\mathcal{N}(\boldsymbol{u})}^{1-\alpha_2}} \sum_{\boldsymbol{v} \in \mathcal{N}(\boldsymbol{u})} \boldsymbol{W}_{\boldsymbol{v}-\boldsymbol{u}} \frac{\boldsymbol{\gamma}}{\hat{\boldsymbol{\sigma}}_{\mathcal{B}}^{\alpha_2}} \quad . \quad (4.8)$$

4.4 Network Architecture

Because we specifically confront the resolution mismatch setting, we use a global 3D architecture for descriptor learning. This is because local 3D descriptors usually crop a region with a fixed radius around the point of interest and extract feature representations based on that region. In the case of resolution mismatch with unknown global scales, regions with a fixed radius may contain dramatically different amounts of the actual, real-world volume. A global 3D architecture is better suited to this resolution mismatch because it has a receptive field that theoretically encompasses the entire volume, regardless of resolution. Intermediate layers in the network can therefore encode both local geometric and global context information of the entire 3D scene. In principle, this allows for encoding similar representations of an object with

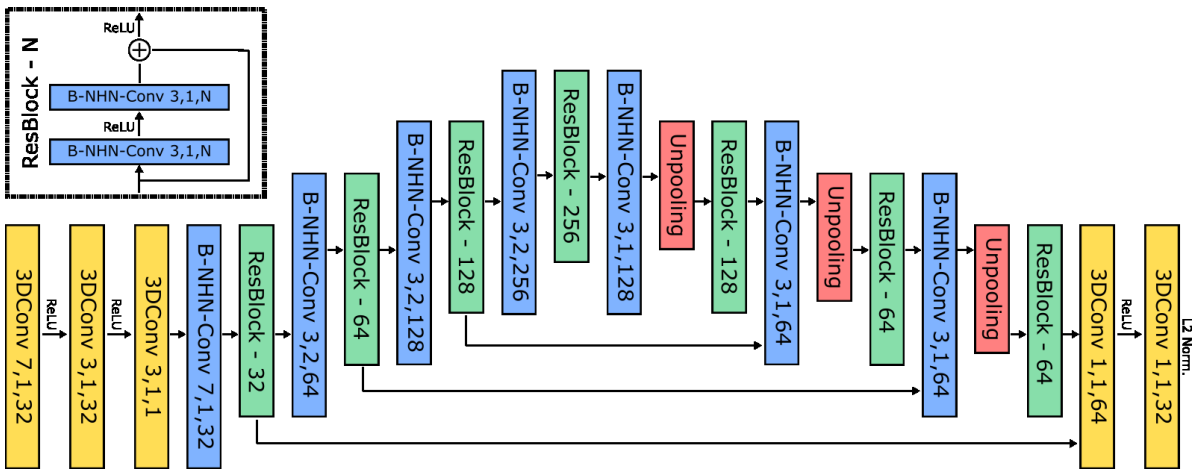

Figure 4.2: **Network architecture for geometric descriptor.** 3DConv stands for sparse 3D convolution, where the three numbers are kernel size along one spatial dimension, stride size, and output channel size. B-NHN-Conv is defined in Eq. 4.6 and the numbers have the same meaning as those in 3DConv. ResBlock is a residual block of B-NHN-Conv layers, shown on the upper-left corner. The skipping arrows mean skip connection with a concatenation operation. Unpooling is a nearest neighbor upsampling operation that copies the value of an input voxel to all occupied ones in a 2x2x2 region of the output volume. We replaced the combination of B-NHN-Conv and Unpooling with the transposed version of B-NHN-Conv, in the experiments of the standard 3DMatch benchmark. In KITTI benchmarks, the kernel sizes of the 1st and 4th layers are 5 instead of 7. In other experiments, we used the one in the figure. When NHN is used instead, all B-NHN-Convs are replaced with NHN-Conv layers.

different resolutions.

FCGF [59] is a voxelized sparse CNN with an encoder-decoder architecture and skipping connections, one of the first methods to apply a global architecture to 3D descriptor learning for sparse correspondence estimation. It achieves superior performance on the 3DMatch and KITTI registration benchmark [136, 148], and so we use the FCGF architecture with modification as the primary backbone for our proposed method. Fig. 4.2 shows the overall network architecture and we refer to this architecture as Mink., short for MinkowskiNet, in the following sections.

4.5 Loss Design

Whereas the original FCGF uses HC loss [59], we use the Relative Response loss in [15]. We observe that the performance of HC loss is much worse in the resolution-mismatch benchmark than in the standard one. This may be due to the need to dynamically adjust the hyperparameters based on the relative resolution for a given sample pair. For each sampled point correspondence in the ground truth, RR loss compares the source point feature embedding with the features of all points in the target sample, maximizing the similarity between features in the ground truth and the estimated point correspondence while minimizing the similarity of all non-corresponding points.

Let $\mathcal{P}_{\mathrm{t}}$ be the set of all point locations in the target sample, $\mathcal{G}$ be a random subset of ground truth point correspondences between source and target sample. Let $\boldsymbol{f}^{\mathrm{s}}_{\boldsymbol{p}_{\mathrm{s}}} \in \mathbb{R}^C$ and $\boldsymbol{f}^{\mathrm{t}}_{\boldsymbol{p}_{\mathrm{t}}} \in \mathbb{R}^C$ be the geometric features at source and target location $\boldsymbol{p}_{\mathrm{s}} \in \mathbb{R}^3$ and $\boldsymbol{p}_{\mathrm{t}} \in \mathbb{R}^3$, respectively. The RR loss is expressed as

$$\mathcal{L}_{\mathrm{rr}} = -\frac{1}{|\mathcal{G}|} \sum_{(\boldsymbol{p}_{\mathrm{s}}, \boldsymbol{p}_{\mathrm{t}}) \in \mathcal{G}} \log \left(\frac{\mathrm{e}^{\sigma \left(\boldsymbol{f}^{\mathrm{s}}_{\boldsymbol{p}_{\mathrm{s}}} \right)^{\intercal} \boldsymbol{f}^{\mathrm{t}}_{\boldsymbol{p}_{\mathrm{t}}}}}{\sum_{\boldsymbol{p} \in \mathcal{P}_{\mathrm{t}}} \mathrm{e}^{\sigma \left(\boldsymbol{f}^{\mathrm{s}}_{\boldsymbol{p}_{\mathrm{s}}} \right)^{\intercal} \boldsymbol{f}^{\mathrm{t}}_{\boldsymbol{p}}}} \right) - \log \left(\frac{|\mathcal{P}_{\mathrm{t}}|}{N} \right) \tag{4.9}$$

where $\sigma \in \mathbb{R}$ is used to enlarge the value range of feature correlation and $N \in \mathbb{R}$ is a constant factor. To account for the fact that the size of the input 3D point cloud could vary, we add a second term, as shown above, to make sure the loss value for samples of various sizes is consistent. This term does not affect the gradient but is important for monitoring progress during training.

4.6 Experiments

We evaluate our approach on a clinical dataset of nasal cavities, as well as on standard benchmark datasets 3DMatch [136] and KITTI [148], in both the standard resolution setting and the resolution mismatch setting. Section 4.6.3 and 4.6.4 provide greater details about the data processing for each benchmark, respectively; here, we provide a brief overview of the experiments and the resolution mismatch setting generated for each. Fig. 4.3 shows sample pairs from each dataset in the case of res-

olution mismatch, with colors representing the learned feature embedding from our descriptor.

We present a dataset of nasal cavity CT volumes to evaluate our NHN-based descriptor for the application of video-CT registration [214]. To produce this data, we have built a statistical shape model of the entire nasal cavity using 52 CT scans collected from The Cancer Imaging Archive [215]. The shape model was generated using the Principal Component Analysis (PCA)-based method in [216]. We generate the resolution mismatch variant of this data by applying a grid downsampling after remeshing, same as the 3DMatch processing below. The data split was based on a different range of mode weights of the shape model in the training, validation, and testing phase.

The 3DMatch [136] dataset contains indoor scenes processed from RGB-D images into point clouds. To generate the resolution mismatch benchmark, we use the same hyperparameters as [136], but instead of sampling points from a TSDF volume, we use the Marching Cubes [119] algorithm to extract a triangular mesh surface, from which we take the vertices as a point cloud. Compared with applying operations such as voxel grid downsampling on point clouds, applying a remeshing operation (*e.g.*, ACVD [217]) on meshes more accurately simulates the resolution variation encountered in practice. This is because it allows for sampling along the surface while preserving mesh geometry. Second, the KITTI odometry dataset [148] depicts out-

door environments, captured using a lidar sensor. Because no mesh surfaces can be extracted from lidar data, we simply used voxel grid downsampling to mimic the resolution variation in the resolution-mismatch benchmark. We split both 3DMatch and KITTI according to the protocol in [59].

4.6.1 Training

For each benchmark, we train the FCGF network using the SGD optimizer with momentum 0.9 and cyclic learning rate within the range of $[1.0\mathrm{e}^{-4}, 7.0\mathrm{e}^{-4}]$. For the RR loss, we set $\sigma = 20$ and, for each sample, use 10 random positive pairs for loss calculation per iteration. For models with B-NHN, all pairs of α_1 and α_2 are initialized with 0.5 in the standard benchmarks. In the resolution-mismatch ones, two-stage training is adopted. All parameters except α_1 and α_2, which are fixed to 0.0, are trained to convergence; the whole network is then jointly fine-tuned. For all experiments, the network is trained until the validation performance plateaus and the batch size is 4. Data augmentation is different for each dataset and is described below for each result.

4.6.2 Evaluation Metrics

We use two common evaluation metrics reported in previous works [59, 205, 218]. To evaluate descriptor performance, we report the feature-match recall for each dataset. Additionally, for the KITTI standard benchmark, we evaluate relative translation and rotation errors after registration.

Feature-match recall. Intuitively, feature-match recall (FMR) measures the percentage of sample pairs where the Random Sample Consensus method [66] can recover the ground truth pose with high confidence. It is defined as

$$\mathrm{FMR} = \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \mathbb{1}\left(I_{\mathrm{s}} > \tau_2\right) \quad \text{, where} \tag{4.10}$$

$$I_{\mathrm{s}} = \frac{1}{N_s} \sum_{(\boldsymbol{x},\boldsymbol{y}) \in \mathcal{M}_s} \mathbb{1}\left(\left\|\boldsymbol{T}^{*}\boldsymbol{x} - \boldsymbol{y}\right\|_2 < \tau_1\right) \quad . \tag{4.11}$$

$\mathcal{S}$ is the set of sample pairs for evaluation. $(\boldsymbol{x}, \boldsymbol{y}) \in \mathbb{R}^3 \times \mathbb{R}^3$ is a pair within the set of the putative 3D point correspondence, $\mathcal{M}_s$, for the sample pair s. $\boldsymbol{T}^{*} \in \mathrm{SE}\left(3\right)$ is the ground truth pose. $\tau_1 \in \mathbb{R}$ is the inlier distance threshold and $\tau_2 \in \mathbb{R}$ is the inlier recall threshold. Same as [59], for each point of the smaller sample in a pair, the one in the other sample that has the most similar feature description is treated as the found correspondence. No further pruning is applied to the correspondence set. Also, following the convention of [59], we use N_s, the number of points of the smaller sample in s. Because $N_s \geq |\mathcal{M}_s|$, this results in a FMR no larger than if Eq. 4.11 were

to have $|\mathcal{M}_s|$ in place of N_s.

Relative translation and rotation error. The relative translation error (RTE) and relative rotation error (RRE) measure the registration errors after RANSAC initialization, using the extracted features. This is an indirect measurement of feature quality, which we report for the KITTI standard benchmark, following common convention. RTE is defined as $\left\|\hat{\boldsymbol{t}} - \boldsymbol{t}^*\right\|_2$, where $\hat{\boldsymbol{t}}$ and $\boldsymbol{t}^*$ are the estimated and ground truth translation, respectively. RRE is defined as $\arccos\left((\mathrm{Tr}(\hat{\boldsymbol{R}}^\mathsf{T}\boldsymbol{R}^*) - 1)/2\right)$, where $\hat{\boldsymbol{R}}$ and $\boldsymbol{R}^*$ are the estimated and ground truth rotation matrices, respectively.

4.6.3 Simulation Study on Nasal Cavity

We evaluate 3D descriptors for the task of video-CT registration [214], using our dataset of nasal cavity volumes. This corresponds to one of our envisioned target applications where the resolution-mismatch problem is inevitable without additional prior knowledge. Because large amounts of ground truth data are difficult to obtain for real nasal cavities, the dataset has been built from simulation using a statistical shape model of the nasal cavity, as described in Sec. 4.6. In video-CT registration, only part of the entire surface in the CT scans can be observed using an endoscope. Therefore, nasal passages were manually segmented from the mean model of the nasal cavity to get the indices of vertices in the statistical shape model that belong to nasal passages. Fig. 4.3 shows an example of a pair of the whole nasal cavity and the

ϕ	$\{1, 1.5, 2\}$			$\{2.5, 3, 3.5\}$			$\{4, 4.5, 5\}$		
τ_2	.30	.50	.70	.30	.50	.70	.30	.50	.70
FCGF [59]	0.125	0.030	0.002	0.543	0.446	0.315	0.775	0.566	0.511
Mink.+BN	0.078	0.016	0.002	0.499	0.475	0.380	0.526	0.500	0.500
Mink.+IN	0.000	0.000	0.000	0.313	0.146	0.000	0.500	0.500	0.435
Mink.+BIN	0.294	0.178	0.109	0.731	0.551	0.496	0.893	0.671	0.535
Mink.+NHN	0.620	0.373	0.204	0.982	0.876	0.630	0.986	0.925	0.777
Mink.+B-NHN	**0.645**	**0.409**	**0.237**	**0.990**	**0.906**	**0.688**	**0.992**	**0.949**	**0.834**

Table 4.1: **Evaluation of geometric descriptors on the dataset of nasal cavity.** For B-NHN, α_1 and α_2 are 0.001 ± 0.064 and 0.055 ± 0.107, respectively. © 2021 IEEE

right nasal passage. During training, full-range rotation and partial cropping with cropping ratio $\in [0, 0.5]$ were applied. As in Sec. 4.6.4, ACVD [217] was applied to remesh the data with random target edge length $\in [2, 20]\,\mathrm{mm}$. As the operation in Sec. 4.6.4, a grid downsampling is applied to the point cloud after remeshing, with a grid size equal to the sampled target edge length. This is to build a sparse 3D volume for Mink. to process. During evaluation, the target edge lengths were $2\,\mathrm{mm}$ and $\phi\,\mathrm{mm}$ for a sample pair. ϕ was set to 9 numbers $\in [1, 5]$. For each ϕ value, 1000 testing sample pairs with random mode weights, full-range rotation, and partial cropping with ratio $\in [0, 0.5]$ were evaluated. We set the inlier distance threshold to $\tau_1 = 4\phi^2\,\mathrm{mm}$ to allow for more error in coarser-resolution samples. Regarding the mode weight sampling, the first 10 mode weights were uniformly sampled $\in [0, 2.5]$, $[2.5, 3]$, and $[-3, 0]$ standard deviation of PCA results during training, validation, and evaluation, respectively. Table 4.1 lists the FMR performance of different methods.

4.6.4 Evaluation on Indoor and Outdoor Dataset

3DMatch standard benchmark. We evaluate our descriptor on the standard 3DMatch benchmark, as in [136], which assumes all samples have the same resolution. To pre-process 3DMatch, point cloud data are first downsampled with a grid size of 5cm. We then apply training augmentations, including full-range random rotation and random scaling $\in [0.8, 1.2]$. In this experiment, we mainly evaluate normalization techniques when used in the state-of-the-art descriptor learning architecture FCGF [59]. These include the commonly used BN [149], IN [152], and BIN [156]; as well as the proposed NHN and B-NHN. We do not include LCN [155] in this analysis because adapting LCN would require the integral image calculation, an intrinsically serial operation [219], of every internal 3D feature map over the sparse volume data, making it impractical in typical applications for point cloud correspondence estimation. In addition to FCGF, we evaluate the performance of other representative 3D architectures [138, 165, 169, 176] which excel at dense prediction tasks like semantic segmentation. Adapting such architectures to make them suitable as 3D descriptors and thereby enable a fair comparison requires some slight modifications, the details of which are provided in Appendix A.

Results are shown in Table 4.2, which lists the FMR for all methods at $\tau_1 = 10\text{cm}$ and $\tau_2 = 0.05$. When using other normalization techniques with Mink., such as BN, IN, and BIN, other than NHN and B-NHN, we substitute the B-NHN-Conv modules

Method	$\tau_2 = .05$	$\tau_2 = .10$	$\tau_2 = .20$	$\tau_2 = .30$
KPConv [169]	0.798	0.517	0.163	0.050
PPNet [220]	0.478	0.250	0.057	0.015
PointNet++ [165]	0.471	0.201	0.026	0.002
DCM-Net [176]	0.001	0.00	0.00	0.00
FCGF [59] (5cm)	**0.935**	0.852	0.613	0.401
Mink.+BN	0.924	0.832	0.588	0.387
Mink.+IN	0.607	0.359	0.136	0.054
Mink.+BIN	0.692	0.422	0.157	0.049
Mink.+NHN	0.866	0.670	0.357	0.166
Mink.+B-NHN	0.933	**0.852**	**0.634**	**0.428**

Table 4.2: **Evaluation of geometric descriptors on the 3DMatch standard benchmark.** All models in this table were trained and evaluated using the point cloud data downsampled with the grid size of $5\,\mathrm{cm}$. The results of FCGF were estimated using the pre-trained model provided by [59]. Note that we did not include the results of FCGF with 2.5cm grid size here, which is the state-of-the-art result reported in [59], for a fair comparison. For B-NHN, α_1 and α_2 are 0.63 ± 0.48 and 0.64 ± 0.31, respectively.

with the normalization module following with a normal 3D convolution for a fair comparison.

3DMatch resolution-mismatch benchmark. In this benchmark, we evaluate the performance of 3D descriptors when a sample pair may contain models with different resolutions. During training, we simulate resolution variation by applying ACVD [217] to meshes with random target edge length $\in [3, 30]\,\mathrm{cm}$. In addition, full-range random rotation was used for training augmentation. Note that in both training and evaluation, the network input is still a point cloud, which consists of vertices of the remeshed sample. All the following experiments involving remeshing

operations also converted remeshed samples to point clouds as network input. To generate a sparse 3D volume as input for voxelized sparse CNN architectures (FCGF and Mink.), we further applied grid downsampling to the point cloud with a grid size equal to the sampled target edge length. This is to simulate the case of resolution mismatch without knowledge of global scales. For methods [165] that use point locations as input features, the samples are further scaled to equalize the average edge lengths to remove the global scale information. The same processing is applied to other resolution-mismatch benchmarks.

During evaluation, the target edge lengths were $3\,\mathrm{cm}$ and $3\phi\,\mathrm{cm}$ for a sample pair. ϕ was set to 9 numbers $\in [1, 5]$; for each ϕ value, all testing sample pairs, same as the standard benchmark, were evaluated. The inlier distance threshold τ_1 is set to $6\phi\,\mathrm{cm}$ to allow for more error in coarser-resolution samples, thus the results for different sets of ϕ are not directly comparable. We did not evaluate local descriptors for reasons mentioned in Sec. 4.4 and the time-consuming data pre-processing procedures that make training infeasible.

Table 4.3 lists the FMR performance of all evaluated methods under various sample resolution ratios ϕ and inlier ratio thresholds τ_2. Note that, in all the resolution-mismatch benchmarks, the results for each set of ϕ are averaged for display purposes. As can be seen, Mink.+B-NHN achieved top performance in all three sets of resolution ratio ϕ. It also shows that PointConv-based methods [169, 220] with averaging

ϕ	$\{1, 1.5, 2\}$			$\{2.5, 3, 3.5\}$			$\{4, 4.5, 5\}$		
τ_2	**.05**	**.10**	**.20**	**.05**	**.10**	**.20**	**.05**	**.10**	**.20**
KPConv [169]	0.121	0.024	0.002	0.015	0.004	0.000	0.034	0.009	0.003
PPNet [220]	0.043	0.007	0.000	0.074	0.014	0.003	0.141	0.032	0.007
PointNet++ [165]	0.004	0.000	0.000	0.014	0.003	0.001	0.042	0.010	0.002
FCGF [59]	0.421	0.220	0.070	0.354	0.112	0.009	0.414	0.156	0.012
Mink.+BN	0.380	0.174	0.046	0.354	0.127	0.015	0.438	0.185	0.024
Mink.+IN	0.206	0.066	0.009	0.271	0.084	0.013	0.362	0.150	0.025
Mink.+BIN	0.106	0.012	0.001	0.168	0.022	0.001	0.257	0.056	0.003
Mink.+NHN	0.468	0.265	0.085	0.497	0.270	0.059	0.528	0.294	0.064
Mink.+B-NHN	**0.494**	**0.289**	**0.106**	**0.521**	**0.308**	**0.091**	**0.579**	**0.366**	**0.104**

Table 4.3: **Evaluation of geometric descriptors on the 3DMatch resolution-mismatch benchmark.** The same operation is also applied to other tables that show results under the resolution-mismatch setting. Here and in the following benchmarks, unless stated otherwise, "FCGF" indicates that we trained the original architecture in [59] with the same setting as Mink.. For B-NHN, α_1 and α_2 are 0.07 ± 0.12 and 0.04 ± 0.15, respectively. © 2021 IEEE

aggregation alone cannot handle this task well.

KITTI standard benchmark. As in [59, 205], we use lidar point cloud data and GPS information provided in the KITTI odometry dataset for 3D descriptor evaluation. In the standard benchmark, a point cloud was downsampled with a grid size of $0.3\,\mathrm{m}$. Training augmentation consisted of a single random scaling $\in [0.8, 1.2]$ per sample pair. For evaluation, we report registration performance and FMR, with $\tau_1 = 0.3\,\mathrm{m}$ and $\tau_2 \in \{0.1, 0.2, 0.3\}$. Table 4.4 lists the performance of 3DFeat [205], FCGF [59] and Mink. with various normalization methods. When evaluating the registration performance, we reduce the maximum times of validation in RANSAC from 10000 in [59] to 1000 for the ease of experiments. This results in a slight increase in error for all

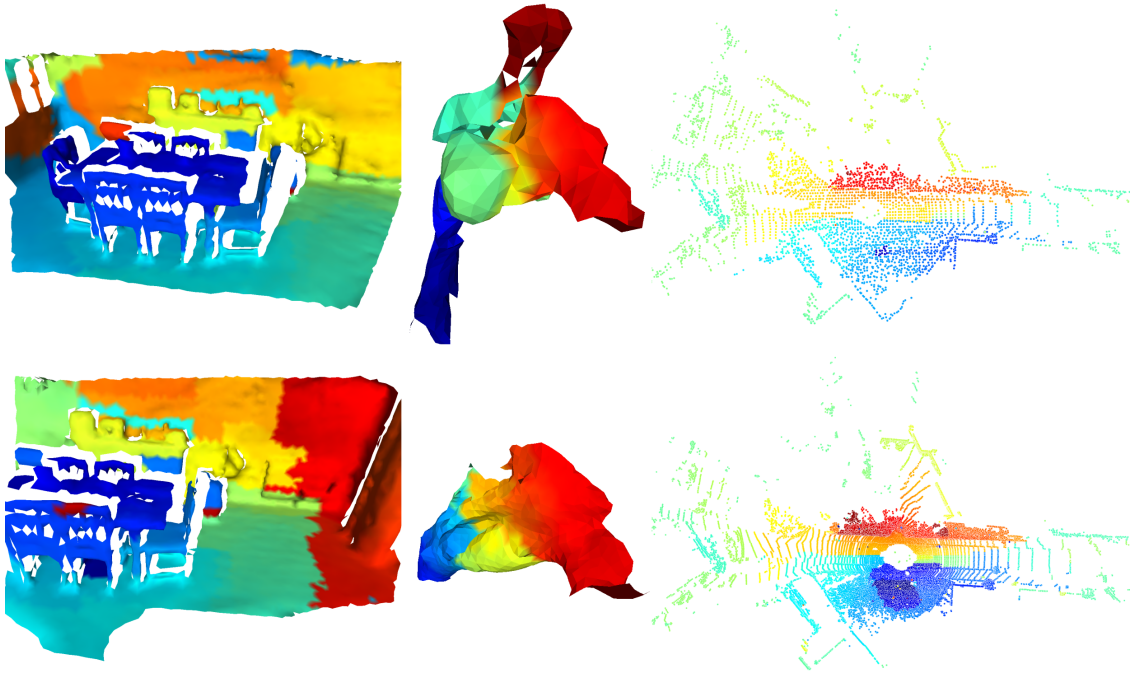

Figure 4.3: **Visualization of geometric feature embeddings with B-NHN.** The displayed sample pairs with mismatched resolutions are from the 3DMatch, clinical, and KITTI datasets. Features were generated by Mink.+B-NHN. For display purposes, each vertex of the meshes in the figure was assigned with the output feature embedding of the spatially closest point in the corresponding point clouds. The feature descriptions were mapped to scalar values with UMAP [221] and displayed in the JET colormap.

Method	*3DFeat	FCGF	Mink.+BN	∼+IN	∼+BIN	∼+NHN	∼+B-NHN
FMR($\tau_2 = 0.1$)	N/A	0.810	0.912	0.908	0.923	0.928	**0.933**
FMR($\tau_2 = 0.2$)	N/A	0.395	0.569	0.589	0.681	0.775	**0.793**
FMR($\tau_2 = 0.3$)	N/A	0.117	0.076	0.065	0.114	0.241	**0.308**
RRE(°)	0.57	0.283	0.234	**0.232**	0.235	0.243	0.244
STD(°)	0.46	0.314	0.236	**0.205**	0.250	0.238	0.248
RTE(cm)	25.90	8.05	6.48	6.51	6.64	6.63	**6.40**
STD(cm)	26.20	7.76	6.13	5.54	6.07	6.36	**5.48**
Success rate(%)	95.97	**99.10**	98.92	98.92	98.74	98.92	98.92

Table 4.4: **Evaluation of geometric descriptors in the KITTI standard benchmark.** The results of 3DFeat are reported in [205]. We evaluated the performance of FCGF using the pre-trained model provided by [59] with grid size $30\,\mathrm{cm}$, the same one as all experiments to its right in this table. STD stands for the standard deviation of the term above it and the symbol "∼" represents "Mink.". For B-NHN, α_1 and α_2 are 0.04 ± 0.04 and 0.06 ± 0.04, respectively. © 2021 IEEE

methods.

As in Table 4.4, Mink.+B-NHN outperforms all comparison methods. The notable FMR performance difference between Mink.+BN and FCGF, we believe, is mainly due to the different loss design, where we use RR loss, described in 4.4, for network training instead of HC loss [59]. Potentially, this could also result from the mild point density variation within a downsampled point cloud, since we observe that HC loss performed inferior to RR loss in all resolution-mismatch benchmarks. Because RANSAC [66] is robust to outliers, as in Table 4.4, the differences of registration performance of all normalization techniques with Mink. are negligible.

KITTI resolution-mismatch benchmark. In this benchmark, we downsampled the point cloud with a random grid size $\in [0.15, 1.5]\,\mathrm{m}$ for training. FMR was used

ϕ	$\{1, 1.5, 2\}$			$\{2.5, 3, 3.5\}$			$\{4, 4.5, 5\}$		
τ_2	.05	.10	.20	.05	.10	.20	.05	.10	.20
FCGF [59]	0.939	0.873	0.371	0.944	0.846	0.410	0.945	0.844	0.450
Mink.+BN	0.915	0.643	0.069	0.927	0.760	0.205	0.941	0.802	0.315
Mink.+IN	0.926	0.796	0.229	0.947	0.869	0.461	0.960	0.905	0.556
Mink.+BIN	0.920	0.706	0.100	0.931	0.778	0.249	0.936	0.820	0.361
Mink.+NHN	0.951	0.919	0.685	0.966	0.929	0.804	0.973	0.934	0.814
Mink.+B-NHN	**0.956**	**0.923**	**0.755**	**0.969**	**0.936**	**0.849**	**0.977**	**0.948**	**0.874**

Table 4.5: **Evaluation of geometric descriptors on the KITTI resolution-mismatch benchmark.** For B-NHN, α_1 and α_2 are 0.01 ± 0.06 and 0.07 ± 0.07, respectively. © 2021 IEEE

as the evaluation metric. During evaluation, the target edge lengths were $0.15\,\mathrm{m}$ and $0.15\phi\,\mathrm{m}$ for a sample pair. ϕ was set to 9 numbers $\in [1, 5]$; for each ϕ value, all testing sample pairs were evaluated. The inlier distance threshold τ_1 is set to $0.3\phi\,\mathrm{m}$ to allow for more error in coarser-resolution samples. Results are shown in Table 4.5.

4.7 Discussion

4.7.1 Connections with Other Normalization

If all convolution operations involved are changed to group-wise convolution, B-NHN-Conv reduces to LCN [155] under the following conditions: 1) the α_1 and α_2 are fixed to zero; 2) the convolution kernel weights $\boldsymbol{W}$ are fixed to the identity mapping of the input voxel centered within the receptive field of the kernel, to the output voxel.

B-NHN-Conv reduces to BN when fixing α_1 and α_2 to one and kernel weights $\boldsymbol{W}$ to the identity mapping. The BN statistics $\boldsymbol{\mu}_{\mathcal{B}}, \boldsymbol{\sigma}_{\mathcal{B}}$ can be replaced with those in IN, LN, *etc*., in which cases B-NHN-Conv reduces similarly.

4.7.2 Limitations

Although NHN and B-NHN perform well on the tasks above, some limitations remain in their use. Compared to global normalization techniques, NHN is sensitive to the sparsity of 3D volume data because of the variability of local neighborhood statistics, an issue that B-NHN mitigates by incorporating a weighted portion of global statistics. This limitation becomes more prominent when applied to sparse voxels, where the local neighborhood is often empty.

We also observe that the optimization of α_1, α_2 in B-NHN is prone to get trapped in local optima. We observe that directly optimizing over all parameters in the mismatch-resolution setting yields worse performance compared with the 2-stage training strategy described in Sec. 4.6.1, a specialized optimization technique could be worth exploring.

For now, we do not have a thorough understanding of why (B-)NHN performs well in the evaluated data variation, resolution mismatch. We aim to investigate more deeply whether the increased performance is indeed caused by the aspects described in Sec. 4.3.1.

In terms of generalizability, we expect that if the resolution mismatch between two samples during evaluation is much different from the training cases, the trained model will not generalize well. If the types of scenes are completely different in training and evaluation, such as nasal cavity for training and indoor scans for evaluation, the trained model will also unlikely generalize well. Therefore, in practice, obtaining a representative collection of training samples that well covers the expected distribution seen during deployment is thus important.

4.8 Conclusion

In this chapter, we aim at the task of global point cloud registration, where we have confronted the challenge of resolution mismatch between samples from different modalities. To do so, we propose a new type of normalization, Batch-Neighborhood Normalization, and show that it increases the robustness of geometric descriptors to point density variation. In empirical experiments, our method surpasses the performance of state-of-the-art models in the resolution mismatch setting and performs favorably in the standard benchmarks. Based on the method design and experiment results, we believe B-NHN is adaptable to other domains that employ convolution, including 2D CNNs and graph neural networks, and is likely suitable for other types of data variation. These areas provide interesting directions to explore for future work.

This work allows for automatic registration between point clouds obtained from different sources because of the proposed network normalization technique. Nevertheless, the normalization itself is general and can be applied to other domains besides the geometric feature descriptor, such as 3D semantic segmentation, 2D image feature descriptor, etc. It would be interesting to see if the robustness to point density mismatch can transfer to other variations, such as image resolution mismatch, illumination changes, and so on, to further improve the network performance in these scenarios. Currently, the geometric features are estimated from point clouds and we did not see improvements with the additional geometry information (*e.g*. surface normal and curvature) that can be obtained from a surface model. However, it is still worthwhile to explore if a different network architecture can exploit such information more effectively and further improve the feature matching performance.

Chapter 5

Real-time Tracking and Reconstruction with Deep Representation

In the previous chapters, we have introduced a complete pipeline for surface reconstruction from a monocular endoscopic video with automatic video-CT registration. The pipeline can provide a high-quality surface model of the anatomy and camera trajectory. However, for some applications (*e.g.*, surgical navigation) in endoscopy, a real-time solution is needed so that the estimate of surface geometry and camera trajectory can provide instant feedback (*e.g.*, regions not yet inspected) to assist the surgeon during the endoscopy procedure. This could also potentially enable au-

tomatic applications such as intelligent endoscope holder and automatic endoscopy inspection.

Simultaneous Localization and Mapping (SLAM) is a type of algorithm that can estimate geometry and trajectory estimates in real-time. Many monocular visual SLAM methods have been developed for general scenes [53, 54, 56, 222–231] and clinical applications such as endoscopy [49, 110, 232–234]. Though such systems have been studied and developed for decades, many practical and theoretical challenges remain. Specifically for endoscopy, scarce texture, illumination variation, tissue deformation, and surgical manipulation are several typical challenges. These challenges either result in low robustness and accuracy of the system running or break certain assumptions of the existing SLAM systems.

In this chapter, we exploit deep learning-based representations to handle the scarce texture and illumination variation, to improve the robustness and accuracy of the system. The deep representation also enables the system to produce surface geometry of the anatomy. Based on our evaluation, the proposed SLAM system generalizes well to unseen endoscopes and subjects and has superior performance compared with a state-of-the-art feature-based SLAM system [54].

CHAPTER 5

5.1 Related Work

5.1.1 Representation Learning for Visual Tracking and Mapping

In recent years, researchers have worked on exploiting prior information learned from previous data to improve the performance of SLAM and Visual Odometry (VO). Different forms of deep depth prior have been used, such as single depth estimate [110, 224, 234], self-improving depth estimate [229], depth estimate with uncertainty [235], and depth estimate with optimizable code [227, 231, 236].

Deep appearance representations have been studied to replace the role of RGB image, which improves convergence basin and enables scenarios with no photometric constancy. BA-Net [236] proposed representation learning with differentiable BA-related loss. DeepSFM [237] extracted implicit feature representation and built cost volume to jointly optimize depth map and relative pose. In this work, we use learning-based viewpoint- and illumination-robust appearance representation and optimizable depth to effectively integrate priors into the SLAM system.

There are also works exploiting other forms of priors for the VO and SLAM systems. For example, Yang *et al*. [235] exploit a pose prior to enable better convergence and mitigate the scale-drift issue; Zhan *et al*. [238] estimate dense optical flow to gain

more robustness towards camera tracking.

5.1.2 Simultaneous Localization and Mapping in Endoscopy

Many SLAM systems have been studied and proposed for the general scene [53, 54, 56, 222–231]. In endoscopy, additional challenges exist compared with other SLAM scenarios such as driving scenes, which are illumination changes, scarce textures, deformation, *etc*.

Feature-based SLAM [49, 109, 239, 240] has been developed for its robustness to illumination changes. However, these systems are not robust to scarce and repetitive textures and thus not suitable for our application. To deal with the scarce texture that causes inaccuracy in terms of trajectory and reconstruction, works have been proposed using either hardware [51] or algorithmic [15, 234, 241] solution. However, the estimated geometry from these systems is not dense and thus cannot allow for target applications of this work that require such information. Deformation happens in endoscopy, especially in certain cases such as laparoscopy and when surgical operations are applied. Works have been developed to confront this challenge [56, 57, 242, 243]. In this work, we exploit learning-based priors and dense geometry to improve the robustness of the system to illumination changes and scarce texture.

5.2 Contributions

In this work, we made the following contributions:

- An effective training scheme to jointly learn optimizable depth and illumination-robust representations with differentiable non-linear optimization.
- A full-feature learning-based dense SLAM system is developed for endoscopy with decent generalizability.
- We demonstrate the effectiveness of the proposed method on *in vivo* and *ex vivo* nasal endoscopic videos, by comparing the performance with state-of-the-art feature-based system ORB-SLAM v3 [54].

From the system point of view, this work presents a real-time option to estimate camera trajectory and dense geometry with a slight sacrifice of accuracy. Many endoscopic applications require such a real-time solution. For example, it enables the surgeon to know which regions have not been observed from the endoscope yet, which increases the chance to find pathology such as polyps, cancer, *etc*. It improves the awareness of surgeons on the critical structures underlying the surface if pre-operative CT imaging is available, which is enabled by the video-CT registration with the dense geometry estimate from the SLAM.

5.3 Representation Learning

In this section, we introduce the learning scheme of various representations used in the proposed SLAM system.

5.3.1 Network Architecture

Two separate networks are used to learn geometry and appearance representations, respectively. In terms of geometry, a depth network is responsible for producing an average depth estimate, which is correct up to a global scale, and a collection of depth bases. The average depth estimate captures the expectation of the depth estimate based on the input color image. However, the task of depth estimation from a single image is ill-posed and therefore errors are expected. The depth bases consist of a set of depth variations that could be used to explain the variation of geometry given the appearance of the input. Therefore, such bases provide a way to further refine the depth estimate, using information from other frames (*e.g.*, geometric consistency), during the optimization process in a SLAM system run.

As shown in Fig. 5.1, the depth network is close to UNet [244] with partial convolution [245]. The endoscope image mask is used in the partial convolutions so that input regions outside the mask do not contribute to the final output. There are two output branches, where one, with absolute function as output activation, predicts the

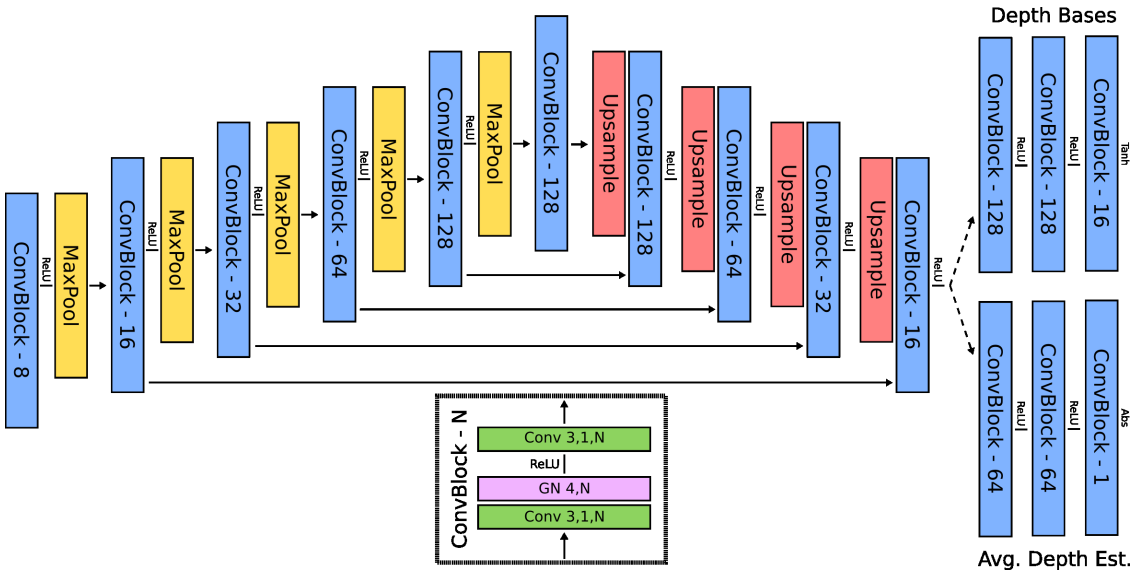

Figure 5.1: **Network architecture for optimizable depth estimation.** Each ConvBlock consists of two partial convolution layers with kernel size as 3 and stride as 1, one group normalization layer with a group size of 4, and one ReLU activation, which are arranged in the way as the figure above. The number after the ConvBlock means the size of the output channel dimension. Two output branches exist in the network for the average depth estimate and the depth bases, described in Sec. 5.3.1. Hyperbolic tangent and absolute functions are used as output activation in these branches.

average depth estimate, and the other produces depth bases with hyperbolic tangent as output activation. The architecture of the discriminator used for depth training is shown in Fig. 5.2.

In terms of appearance, a feature network produces two types of representations. One set of representations, named descriptor map, is used as image descriptors in pair-wise feature matching that are involved in the Reprojection Factor and Sparse Matched Geometry Factor, described in Sec. 5.4.2. A similar training approach as Chapter 2 is used, except that we use point correspondences computed from the sur-

face reconstruction and trajectory instead of the correspondences from SfM. The other set, named feature map, is used for the computation of the Feature-metric Factor as a drop-in replacement of the original video frame. In the image, the illumination of the same location of the scene changes as the viewpoint varies because the lighting source moves with the camera. On the other hand, feature maps can be robust to illumination and viewpoint changes, if the feature network is trained correspondingly.

In this work, we use the task of pair-wise image alignment with differentiable non-linear optimization to train both the appearance and geometry representations, with more details in Sec. 5.3.4. The network architecture for the feature network is the same as the depth network, except for the two output branches. The sizes of channel dimension for the three layers in both the descriptor map and feature map output branches (from hidden to output) are 64, 64, and 16; the output activation functions are both hyperbolic tangent.

5.3.2 Differentiable Optimization

To make the networks learn to master the task of pair-wise image alignment, a differentiable non-linear optimization method is required. In this work, we use Levenberg-Marquardt (LM) algorithm [75] as the optimization solver. LM is a trust-region algorithm to find a minimum of a function over a space of parameters. It is also known as a damped least-squares method because a damping factor is involved in the

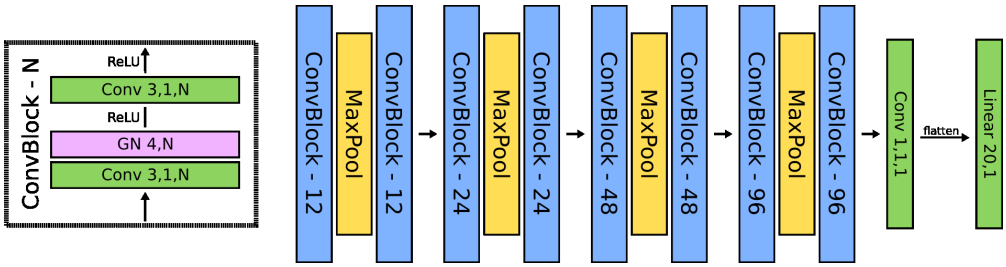

Figure 5.2: **Network architecture of discriminator for depth estimation learning.** The input is the RGB image and the normalized depth map, concatenated along the channel dimension, with a resolution of 64×80. Each ConvBlock consists of two normal convolution layers with kernel size as 3 and stride as 1, one group normalization layer with a group size of 4, and two ReLU activation layers, which are arranged in the way as the figure above. The number after the ConvBlock means the size of the output channel dimension. The final convolution layer, with kernel size as 1, stride as 1, and output channel size as 1, and linear layer, with input channel size as 20 and output channel size as 1, converts the feature map to a scalar value used to indicate the predicted validity of the input sample. Note that before being fed to the final linear layer, the output map from the final convolution layer is first flattened along the sample-wise dimensions.

method that explicitly controls how large the trust region is. The damping factor will decrease or increase based on whether the proposed parameter updates in a single step results in a lower error or not. The larger the damping factor is, the closer LM will be to the gradient descent method. On the other hand, the smaller the damp is, the closer LM will be to the Gauss-Newton method [246], which is a quadratic optimization method.

In the computation graph, all accepted steps are connected, while the accept determination stage and rejected steps are not involved. This removes the need to have an additional network, which is used in BA-Net [236], to predict the damping factor of LM optimization for each iteration, and reduces the complexity of the computa-

tion graph by removing those unnecessary steps. A differentiable solve method for the linear system [82] is used to solve the variable update of a single iteration. With gradient checkpoint technique [82], the number of accepted steps is almost not limited by the memory storage, because each added one will only require a negligible amount of memory space. Therefore, in each iteration of the network training, a long optimization chain can be used.

5.3.3 Loss Design

For each iteration, when the LM optimization converges, several outputs before, during, and after the optimization process will be involved in the loss computation for the network training. Both the average and the optimized depth estimate should agree with the groundtruth depth map up to a global scale. We do not let the depth network try to predict the correct scale and instead leave it to the optimization during SLAM running because predicting a correct depth scale from a monocular endoscopic image is nearly impossible. Therefore, a scale-invariant loss is used for this objective. With a predicted depth map $\boldsymbol{D} \in \mathbb{R}^{1\times H\times W}$, the corresponding groundtruth depth map $\tilde{\boldsymbol{D}} \in \mathbb{R}^{1\times H\times W}$, and the binary video mask $\boldsymbol{V} \in \mathbb{R}^{1\times H\times W}$, the loss is defined as

$$\mathcal{L}_{\text{si}} = \frac{\sum \boldsymbol{D}_{\text{ratio}}^2}{\sum \boldsymbol{V}} + \frac{\left(\sum \boldsymbol{D}_{\text{ratio}}\right)^2}{\left(\sum \boldsymbol{V}\right)^2} \quad , \tag{5.1}$$

where $\boldsymbol{D}_{\text{ratio}} = \log\left(\boldsymbol{V}\boldsymbol{D} + \epsilon\right) - \log\left(\boldsymbol{V}\tilde{\boldsymbol{D}} + \epsilon\right)$. Note all operations, except $\sum$, are element-wise ones; $\sum$ operation sums all elements along the sample-wise dimensions; $\epsilon \in \mathbb{R}$ is a small number to prevent logarithm over zero. Note that all groundtruth data used in this work can be obtained from the surface reconstruction and camera trajectory produced in Chapter 3.

To guide the intermediate depth maps during optimization, we additionally use an adversarial loss. Intuitively, this loss functions as a regularizer and helps make intermediate depth maps more physically feasible given the visual cues (*e.g.*, illumination distribution) in the input color image. This should thus encourage the network to produce a better set of depth bases to produce such depth estimates. The real sample for the GAN will be a color image and the corresponding normalized groundtruth depth map; the fake sample will be the color image and the corresponding normalized depth estimate. For normalization, these depth maps are divided by their maximum value so that the discriminator judges the fidelity of the sample pair based only on the relative geometry and not on the depth scale. The loss form in LS-GAN [247] is used in this work.

For the descriptor map, the RR loss defined in Chapter 2 is used. Because a descriptor map is also used for loop closure detection, besides producing good feature matches on images with large scene overlap, having dissimilar descriptions for images with small or no scene overlap is also desired. A histogram loss is used to make

sure the similarity between histograms of descriptor maps for the source and target images is higher than that for the source and far images. The definitions of these three images are in Sec. 5.3.4. The histogram loss is defined as

$$\mathcal{L}_{\text{hist}} = \frac{1}{C} \sum_{i \in \{1,\dots,C\}} \min \left(\frac{1}{K} d_{\text{EMD}} \left(\boldsymbol{h}_i^{\text{src}}, \boldsymbol{h}_i^{\text{tgt}} \right) - \frac{1}{K} d_{\text{EMD}} \left(\boldsymbol{h}_i^{\text{src}}, \boldsymbol{h}_i^{\text{far}} \right) + \eta_{\text{hist}}, 0 \right) \quad , \tag{5.2}$$

where $d_{\text{EMD}}\left(\boldsymbol{h}_1, \boldsymbol{h}_2\right) = \left\| \text{CDF}\left(\boldsymbol{h}_1\right) - \text{CDF}\left(\boldsymbol{h}_2\right) \right\|_2^2$ measures the earth mover's distance between two histograms. CDF is the operation to produce cumulative density function (CDF) from a histogram. $\boldsymbol{h}_i^{\text{src}} \in \mathbb{R}^K$ is the soft histogram of elements within the valid region of source descriptor map $\boldsymbol{I}^{\text{src}} \in \mathbb{R}^{C \times H \times W}$ along the i^{th} channel, which is $\boldsymbol{I}_i^{\text{src}} \in \mathbb{R}^{1 \times H \times W}$; K is the number of bins in each cumulative density function (CDF) and C is the channel size of the descriptor map; $\eta_{\text{hist}} \in \mathbb{R}$ is a constant margin.

To compute soft CDF differentiably, we refer to the method in [248]. The value of k^{th} bin in the histogram $\boldsymbol{h}_i^{\text{src}}$ can be written as follows

$$\boldsymbol{h}_i^{\text{src}}\left(k\right) = \frac{1}{\left|\Omega^{\text{src}}\right|} \sum_{\boldsymbol{x} \in \Omega^{\text{src}}} \left(\sigma \left(\frac{\boldsymbol{I}_i^{\text{src}}\left(\boldsymbol{x}\right) - \mu_k + 1/K}{\beta} \right) - \sigma \left(\frac{\boldsymbol{I}_i^{\text{src}}\left(\boldsymbol{x}\right) - \mu_k - 1/K}{\beta} \right) \right) \quad , \tag{5.3}$$

where the center value of k^{th} bin is $\mu_k = -1 + \left(2k+1\right)/K \in \mathbb{R}$; the kernel function is $\sigma\left(a\right) = 1/\left(1+\text{e}^{-a}\right)$. The values used in μ_k are related to that the descriptor map has a value range of $(-1, 1)$ because of the architectural design of the feature network. The output activation function is hyperbolic tangent for the descriptor map. Ω^{src} is a

set consisting of all 2D locations within the source video mask; $\beta \in \mathbb{R}$ is a bandwidth parameter. The histograms for target and source images are the same as above except the corresponding descriptor maps are used for calculation instead of the source one.

Intuitively, after the optimization process in Sec. 5.3.2, the source image should be warped to the target frame with good alignment, using the estimate of status. Such a warping process can be described with a 2D scene flow. Therefore, to guide the learning process to produce better image alignment, another loss is to encourage the similarity between the groundtruth 2D scene flow, and the one estimated after the optimization process. This objective will provide signals to both the feature map branch of the feature network and the depth network. This is because a reasonable 2D scene flow can only be achieved if the feature maps are expressive and the depth estimates are accurate, especially when this loss is combined with the depth-related losses above. The flow loss is defines as

$$\mathcal{L}_{\text{flow}} = \frac{1}{\omega^{\text{s}\to\text{t}} \sum \boldsymbol{V}} \sum \boldsymbol{V} \left(\tilde{\boldsymbol{W}}^{\text{s}\to\text{t}} - \boldsymbol{W}^{\text{s}\to\text{t}} \right)^2 \quad , \tag{5.4}$$

where $\tilde{\boldsymbol{W}}^{\text{s}\to\text{t}} \in \mathbb{R}^{2\times H\times W}$ and $\boldsymbol{W}^{\text{s}\to\text{t}} \in \mathbb{R}^{2\times H\times W}$ are the groundtruth and estimated 2D scene flows from source to target frame, respectively. $\omega^{\text{s}\to\text{t}} \in \mathbb{R}$ is a normalization factor, defined as $\omega^{\text{s}\to\text{t}} = \frac{1}{2} \sum \boldsymbol{V} \left((\tilde{\boldsymbol{W}}^{\text{s}\to\text{t}})^2 + (\boldsymbol{W}^{\text{s}\to\text{t}})^2 \right)$. The estimated flow $\boldsymbol{W}^{\text{s}\to\text{t}}$ at

2D location $\boldsymbol{x}^{\mathrm{src}}$ is defined as

$$\boldsymbol{W}^{\mathrm{s}\to\mathrm{t}}\left(\boldsymbol{x}^{\mathrm{src}}\right)=\pi\left(\boldsymbol{p}^{\mathrm{s}\to\mathrm{t}}\right)-\boldsymbol{x}^{\mathrm{src}}\quad\text{, where} \tag{5.5}$$

$$\boldsymbol{p}^{\mathrm{s}\to\mathrm{t}}=\boldsymbol{T}_{\mathrm{src}}^{\mathrm{tgt}}\pi^{-1}\left(\boldsymbol{x}^{\mathrm{src}},\boldsymbol{D}^{\mathrm{src}}\left(\boldsymbol{x}^{\mathrm{src}}\right)\right)\quad. \tag{5.6}$$

$\boldsymbol{p}^{\mathrm{s}\to\mathrm{t}}\in\mathbb{R}^3$ is the 3D location of the lifted source 2D location $\boldsymbol{x}^{\mathrm{src}}\in\mathbb{R}^2$ in the target coordinate system, based on the current estimate of status. π and π^{-1} are the project and unproject operation of the camera geometry. These two operations are the same for all keyframes because camera intrinsics are assumed to be fixed throughout the video. $\boldsymbol{T}_{\mathrm{src}}^{\mathrm{tgt}}=\left(\boldsymbol{T}_{\mathrm{tgt}}^{\mathrm{wld}}\right)^{-1}\boldsymbol{T}_{\mathrm{src}}^{\mathrm{wld}}$ is the relative pose between target and source. $\boldsymbol{D}^{\mathrm{src}}\left(\boldsymbol{x}^{\mathrm{src}}\right)\in\mathbb{R}$ is the depth estimate at 2D location $\boldsymbol{x}^{\mathrm{src}}$ based on the current estimate of depth scale and depth code. It is defined as $\boldsymbol{D}^{\mathrm{src}}\left(\boldsymbol{x}^{\mathrm{src}}\right)=s^{\mathrm{src}}\left(\bar{\boldsymbol{D}}^{\mathrm{src}}\left(\boldsymbol{x}^{\mathrm{src}}\right)+\left(\boldsymbol{c}^{\mathrm{src}}\right)^{\intercal}\hat{\boldsymbol{D}}^{\mathrm{src}}\left(\boldsymbol{x}^{\mathrm{src}}\right)\right)$. The source average depth estimate and depth bases are $\bar{\boldsymbol{D}}^{\mathrm{src}}\in\mathbb{R}^{1\times H\times W}$ and $\hat{\boldsymbol{D}}^{\mathrm{src}}\in\mathbb{R}^{B\times H\times W}$. the source depth scale, depth code, and camera pose matrix are $s^{\mathrm{src}}\in\mathbb{R}$, $\boldsymbol{c}^{\mathrm{src}}\in\mathbb{R}^B$, and $\boldsymbol{T}_{\mathrm{src}}^{\mathrm{wld}}\in\mathrm{SE}\left(3\right)$, respectively. Note that the forms of all definitions related to the other images are the same as the source, except that the superscript symbol should be changed correspondingly.

5.3.4 Training Procedure

In each iteration, three images are used for training, which are the source, target, and far images. Source and target are two images with a large scene overlap, while the far image has a small or no scene overlap with the source. For the source and target images, the depth network produces the average depth estimate and depth bases, and the feature network produces a feature map and descriptor map. The far image is only used in the second-stage training described later and is only involved in the loss calculation for the descriptor map from the feature network.

The network training consists of two stages. At the first stage, the depth estimates (excluding the depth bases) and the descriptor maps (excluding the feature map) are trained separately with the scale-invariant loss and RR loss, respectively. After both networks are trained to a reasonable state, the training moves to the second stage, where two networks are jointly trained with the scheme below. The objective then becomes that, with good geometry and appearance representations produced from these two networks, a source image should be well aligned to a target image with a non-linear optimization. The variables that are optimized over are relative camera pose, depth scale, and depth code associated with the source image. The factors involved are pair-wise factors, FM, SMG, and GC, and prior factors, SC and CD. A random relative camera pose and all-zero depth code are initialized. The initial source depth scale is computed so that the mean values of target and source depth maps are equal. After

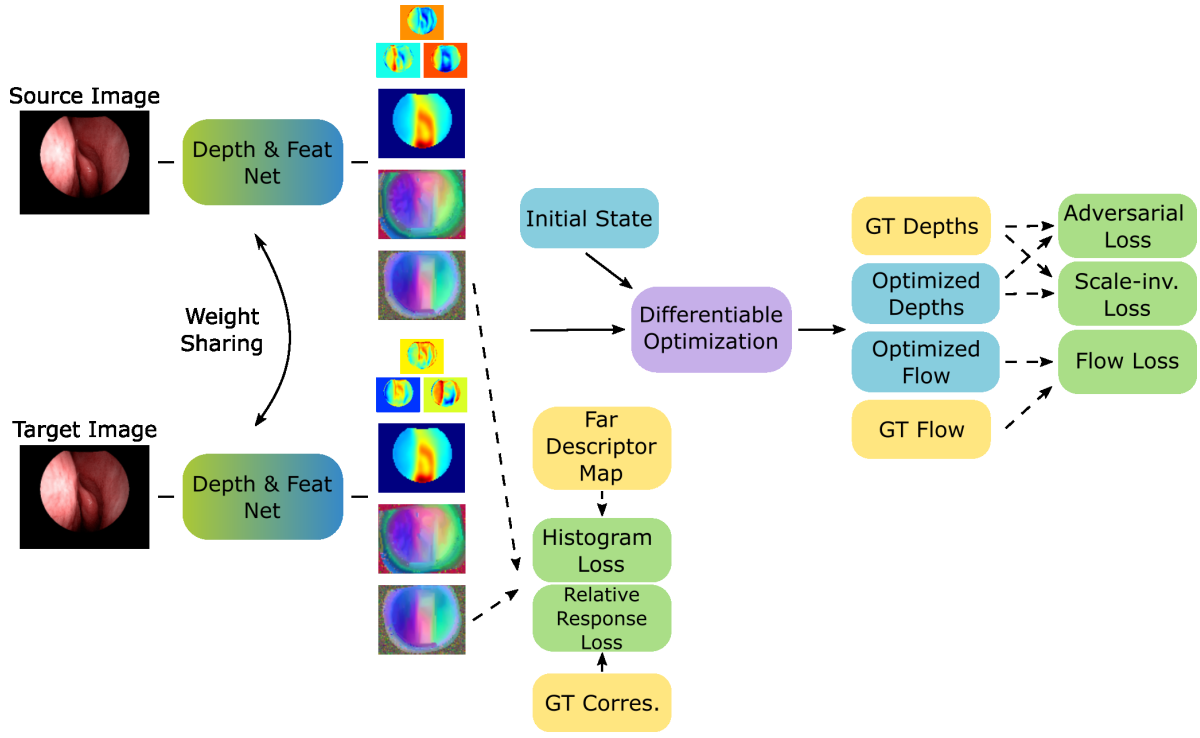

Figure 5.3: **Diagram of representation learning.** The network outputs for each image (from top to bottom) are the depth bases, average depth estimate, feature map, and descriptor map. Network outputs and the initial state of variables (relative pose, depth scales, depth codes) are input to the differentiable optimization pipeline to obtain optimized depth estimates and 2D flow map for loss computation. Descriptor map for the far image is used in the histogram loss. More details are described in Sec. 5.3.

these pre-processing, the optimization described in Sec. 5.3.2 is applied to minimize the objective described by the factors.

With the optimization finished, the loss functions described in Sec. 5.3.3 are calculated and the networks then get updated. Note that there is also a typical GAN-related training cycle [247] involved because we use the adversarial loss for depth training. The training diagram for the second stage is shown in Fig. 5.3.

5.4 Simultaneous Localization and Mapping

5.4.1 Overview

The SLAM system modules are organized into frontend and backend threads. Frontend consists of *Camera Tracking* and *Keyframe Creation* modules. The *Camera Tracking* module is used to track the new video frame against the reference keyframe, where the depth scale of the new frame and relative pose will be optimized over. The *Keyframe Creation* module is used to determine if a new keyframe is needed. If so, a new keyframe will be created and the connections to temporally close keyframes will be built. For each keyframe, a bag-of-words description will be created for efficient global loop detection later in the *Loop Closure* module.

Backend threads run *Loop Closure* and *Mapping* modules. The *Loop Closure* module constantly detects both local and global connections between all keyframe pairs. Whenever a global connection is detected, a lightweight pose-scale graph optimization will be applied to close the loop by adjusting depth scales and camera poses. The *Mapping* module runs full factor graph optimization constantly, where all depth codes, depth scales, and camera poses are jointly optimized with all factors that are described in Sec. 5.4.2. The overall diagram of the SLAM system is shown in Fig. 5.4.

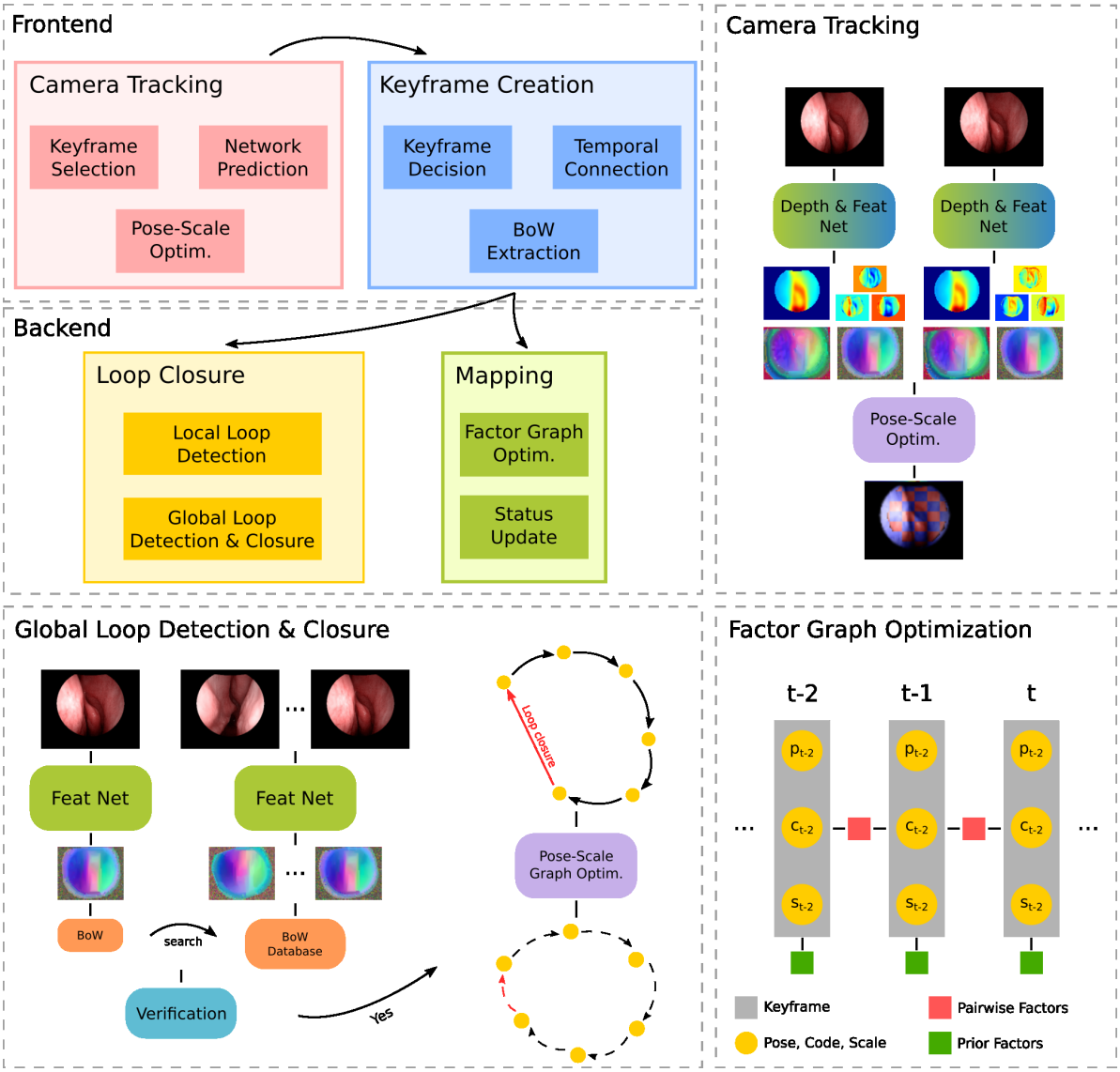

Figure 5.4: **Overall diagram of SLAM system.** The top left shows the module relationship in the proposed SLAM system. The top right demonstrates the network prediction and pose-scale optimization within the *Camera Tracking* module. Note that only a subset of depth bases is displayed. The bottom left shows the process of global loop detection and closure of the *Loop Closure* module. The bottom right demonstrates factor graph optimization in the *Mapping* module. In the pose-scale graph optimization of the global loop closure, only poses and depth scales are optimized. Note that pair-wise factors only between adjacent keyframes are displayed for simplicity.

5.4.2 Factor Design

Feature-metric Factor. Intuitively, if all relevant variables are accurate, the source image warped to the target image plane should align well with the target image in terms of appearance. In this factor, the feature map from the feature network is used as the appearance representation of a frame for reasons described in Sec. 5.3.1. The feature map is pre-processed to form a Gaussian pyramid with a specified number of levels to increase the convergence basin. To build a certain level of the Gaussian pyramid, the Gaussian smoothing operation with a specified size and sigma, and 2-time downsampling will be applied sequentially to the map in the previous level. Note that the binary endoscope mask is also used in generating the Gaussian pyramid so that invalid regions do not contribute to the Gaussian smoothing.

The source feature map pyramid is defined as $\mathcal{F}^{\text{src}} = \{\boldsymbol{F}_i^{\text{src}} | i = 1, \dots, L\}$, where L is the number of levels and $\boldsymbol{F}_i^{\text{src}} \in \mathbb{R}^{C \times H/2^{i-1} \times W/2^{i-1}}$ is the feature map at pyramid level i; The objective of this factor is defined below.

$$\mathcal{L}_{\text{fm}} = \frac{1}{L} \sum_{i=1}^{L} \frac{1}{|\Omega_{\text{src,tgt}}|} \sum_{\boldsymbol{x}^{\text{src}} \in \Omega_{\text{src,tgt}}} \left\| \boldsymbol{F}_i^{\text{tgt}} \left(\pi \left(\boldsymbol{p}^{\text{s} \to \text{t}} \right) \right) - \boldsymbol{F}_i^{\text{src}} \left(\boldsymbol{x}^{\text{src}} \right) \right\|_2^2 \quad , \tag{5.7}$$

where $\Omega_{\text{src,tgt}}$ is the set of source 2D locations that can be projected onto the target mask region given the estimate of the status.

Sparse Matched Geometry Factor. In cases where variables of two frames

are far from being accurate, it is difficult to rely only on the Feature-metric Factor to converge to the correct optimization minima. This is because, even though the feature network is trained to produce feature maps with a better convergence ability, it still has the issue of a relatively small convergence basin, which is common for the appearance-warping based objectives [101].

The descriptor map from the feature network can estimate 2D point correspondences between images through pair-wise feature matching described in Sec. 2.4.3. This enables the objective to have global convergence characteristics. Because in this work, each keyframe has a depth estimate, we can extend the 2D correspondences to 3D ones. Compared with 2D ones, this results in fewer outliers in the correspondences after the geometric outlier removal, which follows the feature matching process. It is because we have depth information available and the outlier removal based on point cloud alignment has less ambiguity than the 2D filtering method based on epipolar geometry. Intuitively, if the depth estimates of two keyframes are correct up to a global scale, a similarity transform estimated from the inlier matches should align two point clouds well. After alignment, outlier matches are those whose spatial distances are larger than the corresponding noise bounds.

The point cloud registration used in this work is Teaser++ [196], which is shown to be robust to a large outlier rate. Teaser++ originally allows single noise bound and we extend its implementation so that a point-wise noise bound can be used. For

each feature match, we set the noise bound to be the depth value of the matched point in the target image multiplying a specified constant factor. Geometrically, this corresponds to how many pixels are allowed in the location difference between the matched and projected 2D location. The definition of this factor is:

$$\mathcal{L}_{\text{smg}} = \frac{1}{|\mathcal{M}|} \sum_{(\boldsymbol{x}^{\text{src}}, \boldsymbol{x}^{\text{tgt}}) \in \mathcal{M}} \rho_{\text{fair}} \left(\left\| \boldsymbol{p}^{\text{s}\to\text{t}} - \pi^{-1} \left(\boldsymbol{x}^{\text{tgt}}, \boldsymbol{D}^{\text{tgt}} \left(\boldsymbol{x}^{\text{tgt}} \right) \right) \right\|_2^2 ; \delta_{\text{smg}}^{\text{src}} \right) \quad , \tag{5.8}$$

where $\mathcal{M}$ is a set of feature matches consisting of pairs of 2D locations $(\boldsymbol{x}^{\text{src}}, \boldsymbol{x}^{\text{tgt}}) \in \mathbb{R}^2 \times \mathbb{R}^2$, and $\delta_{\text{smg}}^{\text{src}} = \frac{\sigma_{\text{smg}}}{|\Omega^{\text{src}}|} \sum_{\boldsymbol{x} \in \Omega^{\text{src}}} \bar{\boldsymbol{D}}^{\text{src}} (\boldsymbol{x})$, which is the mean value of the source average depth estimate multiplying a constant factor $\sigma_{\text{smg}} \in \mathbb{R}$. The outlier-robust "Fair" loss [246] is used, which is defined as $\rho_{\text{fair}} (a; b) = 2(\sqrt{a/b} - \ln (1 + \sqrt{a/b}))$.

Reprojection Factor. This factor behaves similarly to the Sparse Matched Geometry Factor except that the objective is changed from minimizing the average distance of 3D point sets to minimizing the average distance of projected source-to-target 2D locations and target 2D locations. The factor is defined as:

$$\mathcal{L}_{\text{rp}} = \frac{1}{|\mathcal{M}|} \sum_{(\boldsymbol{x}^{\text{src}}, \boldsymbol{x}^{\text{tgt}}) \in \mathcal{M}} \rho_{\text{fair}} \left(\left\| \pi \left(\boldsymbol{T}_{\text{src}}^{\text{tgt}} \pi^{-1} \left(\boldsymbol{x}^{\text{src}}, \boldsymbol{D}^{\text{src}} \left(\boldsymbol{x}^{\text{src}} \right) \right) \right) - \boldsymbol{x}^{\text{tgt}} \right\|_2^2 ; \sigma_{\text{rp}} W^2 \right) \quad , \tag{5.9}$$

where $\sigma_{\text{rp}} \in \mathbb{R}$ is a multiplying factor and W is the width of the involved depth map. In this work, we assume all keyframes have the same resolution.

Geometric Consistency Factor. In all factors above, only one depth estimate

of the image pair is used, except the Sparse Matched Geometry Factor that is only used in the geometric verification described in Sec. 5.4.6. Therefore, the geometric consistency between two depth estimates is not enforced yet. On the other hand, this factor ensures such consistency by encouraging the source depth estimate transformed to the target coordinate to have consistent values as the target depth estimate. The factor is defined as:

$$\mathcal{L}_{\mathrm{gc}} = \frac{1}{|\mathcal{M}|} \sum_{(\boldsymbol{x}^{\mathrm{src}}, \boldsymbol{x}^{\mathrm{tgt}}) \in \mathcal{M}} \rho_{\mathrm{cauchy}} \left(\left\| z^{\mathrm{s}\to\mathrm{t}} - \boldsymbol{D}^{\mathrm{tgt}} \left(\pi \left(\boldsymbol{p}^{\mathrm{s}\to\mathrm{t}} \right) \right) \right\|_2^2 ; \delta_{\mathrm{gc}}^{\mathrm{src}} \right) \quad , \tag{5.10}$$

where $z^{\mathrm{s}\to\mathrm{t}}$ is the z-axis component of $\boldsymbol{p}^{\mathrm{s}\to\mathrm{t}}$; $\delta_{\mathrm{gc}}^{\mathrm{src}}$ is the same as $\delta_{\mathrm{smg}}^{\mathrm{src}}$, except that σ_{gc} is used instead of σ_{smg}. Cauchy loss [246] is used to increase the robustness of this factor, which is defined as $\rho_{\mathrm{cauchy}}\left(a; b\right) = \ln\left(1 + a/b\right)$.

Relative Pose Scale Factor. This factor is used only in the pose-scale graph optimization over depth scales and camera poses for the global loop closure described in Sec. 5.4.6. The intuition of this factor is the scale ambiguity of pair-wise factors for a keyframe pair. The error value for the pair-wise factors above will not change if depth scales and the translation component of the relative camera pose are scaled jointly. In the stage of global loop closure, all frame pairs except the newly detected global loop should have reasonably variable estimates. Therefore, the functionality of this factor is to keep variable estimates in the previous links as close to the original

estimates as possible up to a global scale and make the new global loop pair reach the goal. Specifically, the overall objective is to make the ratio of depth scales, rotation of the relative pose, and translation of the relative pose up to a scale reach the target values. The factor is defined as follows:

$$\begin{aligned}\mathcal{L}_{\text{rps}} = &\left\| \frac{\boldsymbol{t}_{\text{src}}^{\text{tgt}}}{s^{\text{src}}} - \frac{\tilde{\boldsymbol{t}}_{\text{src}}^{\text{tgt}}}{\tilde{s}^{\text{src}}} \right\|_2^2 + \omega_{\text{rot}} \left\| \log\left(\boldsymbol{R}_{\text{src}}^{\text{tgt}}\right) - \log\left(\tilde{\boldsymbol{R}}_{\text{src}}^{\text{tgt}}\right) \right\|_2^2 + \\ &\omega_{\text{scl}} \left(\log\left(\frac{s^{\text{tgt}}}{s^{\text{src}}}\right) - \log\left(\frac{\tilde{s}^{\text{tgt}}}{\tilde{s}^{\text{src}}}\right) \right)^2 \quad ,\end{aligned} \tag{5.11}$$

where $\boldsymbol{t}_{\text{src}}^{\text{tgt}} \in \mathbb{R}^3$ and $\boldsymbol{R}_{\text{src}}^{\text{tgt}} \in \text{SO}(3)$ are the translation and rotation components of the relative pose $\boldsymbol{T}_{\text{src}}^{\text{tgt}}$ described above, respectively. Note that the logarithm operation on the rotation components is the matrix logarithm of $\text{SO}(3)$ [249]. $\omega_{\text{rot}} \in \mathbb{R}$ and $\omega_{\text{scl}} \in \mathbb{R}$ are the weights for the rotation and scale components of this factor, respectively. In this equation and the ones below, every symbol with $\tilde{}$ on top represents the target counterpart of the one without it.

Code Factor. This factor is used to keep the depth code of a keyframe within a reasonable range. Each keyframe has this factor included in the full factor graph. Note that this factor and the following Scale Factor and Pose Factor only involve one keyframe per factor. It is defined as

$$\mathcal{L}_{\text{code}} = \frac{1}{B} \left\| \boldsymbol{c}^{\text{src}} - \tilde{\boldsymbol{c}}^{\text{src}} \right\|_2^2 \quad . \tag{5.12}$$

Scale Factor. This factor is to make the depth scale of a keyframe as close to the target scale as possible. It is used for the first keyframe in the full factor graph and the new global loop pair in the pose-scale graph of the process of global loop closure. It is defined as

$$\mathcal{L}_{\text{scale}} = \left(\log\left(s^{\text{src}}\right) - \log\left(\tilde{s}^{\,\text{src}}\right)\right)^2 \quad . \tag{5.13}$$

Pose Factor. This factor is used for the first keyframe to anchor the pose trajectory of the entire graph. It is defined as

$$\mathcal{L}_{\text{pose}} = \left\|\boldsymbol{p}_{\text{src}}^{\text{wld}} - \tilde{\boldsymbol{p}}_{\,\text{src}}^{\,\text{wld}}\right\|_2^2 + \omega_{\text{r}} \left\|\log\left(\boldsymbol{R}_{\text{src}}^{\text{wld}}\right) - \log\left(\tilde{\boldsymbol{R}}_{\text{src}}^{\text{wld}}\right)\right\|_2^2 \quad , \tag{5.14}$$

where $\omega_{\text{r}} \in \mathbb{R}$ is the weight of the rotation component of this factor.

5.4.3 Camera Tracking

This module is used to continuously track new video frames to provide a good initialization point for the other modules, shown in Fig. 5.4. When a new frame comes in, it will be tracked against the reference keyframe. The spatially closest keyframe against the latest tracked frame is used as the reference, where the distance is based on the current estimates of camera poses. In some cases, the selection could be wrong because of drifting errors, especially when it is temporally far from the latest keyframe. To verify the selection, the feature matching inlier ratio between

the new frame and the selected keyframe is computed as in the Reprojection Factor. The same metric is also computed between the new frame and the latest keyframe. If the former is smaller than the latter multiplying a factor, the latest keyframe will be used instead as the reference.

Camera tracking is solved with LM optimization over the relative camera pose between the new frame and reference and the depth scale of the new frame. The factors involved are the Feature-metric Factor and Reprojection Factor. The termination of optimization is based on several criteria, which are the maximum number of iterations, parameter update ratio threshold, and gradient threshold. In this module, only the relative pose $\boldsymbol{T}_{\text{src}}^{\text{tgt}}$ is optimized over. Once the optimization finishes, the pose of the new frame, labeled as source, can be calculated as $\boldsymbol{T}_{\text{src}}^{\text{wld}} = \boldsymbol{T}_{\text{tgt}}^{\text{wld}} \boldsymbol{T}_{\text{src}}^{\text{tgt}}$, where $\boldsymbol{T}_{\text{tgt}}^{\text{wld}}$ is the camera pose of the reference keyframe.

5.4.4 Keyframe Creation

This module is for handling keyframe creation and pre-processing. When the first keyframe is created, the depth scale is initialized so that the median value of the average depth estimate is set to one. In this way, the global scale of the camera trajectory is relatively stable. And thus the values of different components in the Relative Pose Scale Factor, used in the *Loop Closure* module, are stable across different videos and different trained models with a fixed parameter setting. This, in turn, results in more

stable global loop closure performance. In terms of the prior factors, for the first keyframe, all three prior factors, *i.e.*, Code Factor, Scale Factor, and Pose Factor, are integrated into the factor graph, while, for the other keyframes, only Code Factor will be constructed.

For every tracked new frame, this module determines if a new keyframe is needed. Because the global scale of the entire graph is ambiguous, no absolute distance threshold can be relied on. Instead, we use a set of more intuitive criteria that directly relate to the information gain of a new frame, which are scene overlap, feature match inlier ratio, and the average magnitude of 2D scene flow. Scene overlap measures the overlap between two frames and reflects how much new region is observed from a new frame. Feature match inlier ratio is the ratio of inlier matches over all the feature match candidates. This reflects how dissimilar the two frames are in terms of appearance, which may be due to a small region overlap, a dramatic texture change, *etc*. As for the texture change, it could be caused by auto exposure adjustment, tissue bleeding, and so on. The average magnitude of 2D scene flow measures how much movement the content of a frame has. It measures one additional movement that is the in-plane camera rotation. This is to track the camera movement of keyframes more continuously and to produce more consistent descriptors and feature maps between keyframes.

For each keyframe, a bag-of-words description is computed from the descriptor

map and added to the database for global loop indexing, described in Sec. 5.4.6. Temporal connections will be added to the new keyframe. These only consist of temporally close keyframes. The number of temporally connected keyframes depends on the feature match inlier ratio. At least one keyframe will be connected to the new one. Additional keyframes, up to a specified maximum number, will be connected only if the ratio between the additional keyframe and the new keyframe is larger than a specified threshold. The factors involved in the pair-wise keyframe connections are the Feature-metric Factor and Geometric Consistency Factor.

5.4.5 Mapping

Mapping is constantly running at the backend. The framework for factor graph optimization is ISAM2 [250]. The entire factor graph consisting of pair-wise and prior factors from all keyframes is optimized in this module, where Fig. 5.4 shows an example of the factor graph. The variables jointly optimized are camera poses, depth scales, and depth codes of all keyframes. Whenever a global loop closure in the *Loop Closure* module finishes, all involved variables in the full factor graph will be reinitialized with the new values.

5.4.6 Loop Closure

As another backend module, the *Loop Closure* module constantly tries to find potential keyframe pairs that can be local or global loop connections and handles the closure correspondingly. For local loop detection, the keyframes, which are visited before the query one within a specified temporal range, are searched. Because the temporal window is set to be small, the trajectory drifting error will not be large, the camera pose of each keyframe can still be roughly relied on for filtering candidates. For this reason, the spatial distance between keyframe pairs is first used.

For the following verification steps, the query keyframe and the closest one within its temporal connections are used as the reference pair. If the spatial distance between the candidate pair is smaller than the spatial distance between the reference pair multiplying a constant factor, the pair will be kept. For pairs being kept after distance filtering, the appearance verification will be run, where the feature match inlier ratio is computed. The candidate pair will be kept if the inlier ratio is larger than that of the reference pair multiplying a constant factor and a specified constant inlier ratio. Lastly, a geometric verification is applied, where a pair-wise optimization similar to the one in the *Camera Tracking* module is run. The difference in terms of factors is that the Sparse Match Geometry Factor is used in place of the Reprojection Factor. It is because the Sparse Match Geometry Factor optimizes 3D distances instead of 2D ones and therefore has higher robustness on variable initialization.

The local connection will only be accepted if the overlap ratio and flow magnitude, computed in the geometric verification, are larger and smaller than those of the reference pair multiplying a constant factor, respectively. After verification, only the best candidate left, in terms of overlap ratio and flow magnitude, will be used to build the local connection. The selected keyframe pairs are linked with pair-wise factors same as the temporal connections.

Another part of this module is global loop connection and closure, as shown in Fig. 5.4. Global loop detection searches for keyframe pairs whose interval is beyond a specified temporal range. Unlike the local loop detection where camera poses can still be relied on to choose candidates, global loop detection uses the appearance of keyframes for the initial candidate selection. The descriptor map estimated by the feature network per keyframe describes the point-wise appearance distinctively and is suitable to be used as the representation to build a bag-of-words place recognition model [251].

A hierarchical bag-of-words method [251] is used in this work, where the model is built from the estimated descriptor maps of a training dataset. Whenever a keyframe is created, the bag-of-words descriptor will be added to a database. When a global loop connection is searched for a query keyframe, the database will be searched through with the extracted bag-of-words descriptor. A specified number of keyframes that are similar to the query keyframe in terms of bag-of-words descriptor will be selected

as candidates. The candidates are then filtered so that the description similarities between the query keyframe and candidates are larger than the similarity between the reference pair multiplying a specified constant factor. One additional requirement is that candidates should not be temporally close to the query keyframe, opposite to the local loop connection. After that, the same appearance and geometric verification as the local loop detection are used to verify the global loop candidates. The verified candidates are ranked based on feature match inlier ratio and, from high to low, each candidate that is temporally far enough from the selected candidates is added to avoid connection redundancy.

Unlike the local loop connection, for the global one, the drifting error between the global keyframe pair is often large. Therefore, it is slow to rely on the full graph optimization in the *Mapping* module to close the gap. To this end, we design a lightweight pose-scale graph optimization for the global loop closure, where the camera poses and depth scales of all keyframes are optimized jointly. In this graph, a set of lightweight factors are used. For the new global loop pair, the Scale Factor and Relative Pose Scale Factor are used, where the target depth scales come from the geometric verification above; For all other keyframe connections, the Relative Pose Scale Factor is used, where the current values are used as the target scales and poses in the factors. The graph optimization terminates if one of two conditions is met: 1) the number of iterations reaches a specified number and 2) the number of consecutive iterations

with no relinearization reaches a specified number. After the optimization finishes, depth scales and camera poses of all keyframes, in the full factor graph of the *Mapping* module, are reinitialized correspondingly with the estimates from the pose-scale graph.

5.5 Experiments

5.5.1 Experiment Setup

The endoscopic videos used in the experiments were acquired from seven consenting patients and four cadavers under an Institutional Review Boards (IRB)-approved protocol. The anatomy captured in the videos is the nasal cavity. The total time duration of videos is around 40 minutes. The input images to both networks are 8-time spatially downsampled, resulting in a resolution of 128×160; the output maps of both networks have a resolution of 64×80. Note that the binary masks with the same resolution are also fed, together with images, into the networks to exclude contributions of invalid pixels. SGD optimizer with cyclic learning rate scheduler [84] is used for network training, where the learning rate range is $[1.0\mathrm{e}^{-4}, 5.0\mathrm{e}^{-4}]$. The weights for scale-invariant loss, RR loss, flow loss, histogram loss, generator adversarial loss, and discriminator adversarial loss are 20.0, 4.0, 10.0, 4.0, 1.0, and 1.0. In terms of the

hyperparameters related to loss design, ϵ is $1.0\mathrm{e}^{-4}$; η_{hist} is 0.3; β is $\frac{4}{5K}$; K is 100; C is 16; H is 64; W is 80; B is 16;

Full-range rotation augmentation is used for input images to the networks during training. The first stage of training lasts for 40 epochs and the second stage lasts until the loss curves plateau, where each epoch consists of 300 iterations with the batch size of 1. Image pairs are selected so that the groundtruth ratio of scene overlap is larger than 0.6; the initialized relative pose is randomized so that the initial ratio of scene overlap is larger than 0.4.

In terms of the hyperparameters of the differentiable LM optimization, damp value range is $[1.0\mathrm{e}^{-6}, 1.0\mathrm{e}^{-2}]$, with $1.0\mathrm{e}^{-4}$ as the initial value. The increasing and decreasing multiplier of the damp value is 11.0 and 9.0, respectively. LM optimization terminates when one of the three below is met: 1) number of iterations reaching 40, 2) maximum gradient smaller than $1.0\mathrm{e}^{-4}$, 3) maximum parameter increment ratio smaller than $1.0\mathrm{e}^{-2}$. Factors involved have the same parameter setting as the SLAM system, which will be described below.

Below are the hyperparameters of the SLAM system. For the *Camera Tracking* module, the multiplying factor used for the reference keyframe selection is 0.6; the maximum number of iterations in the optimization is 40; the damp value range is $[1.0\mathrm{e}^{-6}, 1.0\mathrm{e}^{-2}]$, with $1.0\mathrm{e}^{-4}$ as the initial value; the increasing and decreasing multiplier is 100.0 and 10.0, respectively; the jacobian matrix recompute condition is when

the error update between steps is larger than $1.0\mathrm{e}^{-2}$ of the current error. As for factors in the *Camera Tracking* module, settings are as follows. In the Feature-metric Factor, all samples within the video mask are used for computation; the weights for all 4 pyramid levels (from high resolution to low one) are 10.0, 9.0, 8.0, and 7.0. In the Reprojection Factor, the factor weight and σ_{rp} are 0.1 and 0.03, respectively. In the Sparse Matched Geometry Factor, the factor weight and σ_{smg} are 0.1 and 0.1, respectively; the number of feature match candidates before filtering is 256; in terms of the Teaser++ filtering, the maximum clique time limit, rotation maximum iterations, rotation graph, inlier selection mode, and noise bound multiplier are $50\mathrm{ms}$, 20, chain mode, no inlier selection, and 2.0, respectively; Other parameters of Teaser++ are set to the default ones.

For the *Keyframe Creation* module, settings are as below. The maximum ratios of scene overlap in terms of the area and the number of point inliers within the video mask for a new keyframe are 0.8 and 0.9, respectively; the maximum feature match inlier ratio is 0.4; the minimum average magnitude of 2D flow is 0.08 of the image width. For the temporal connection building in the *Keyframe Creation* module, the maximum number of temporal connections per keyframe is 3; the minimum feature match inlier ratio to connect a previous keyframe is 0.7.

For the *Loop Closure* module, settings are shown as follows. For the local loop detection, the temporal window for searching is 9; the rotation and translation weights

to compute pose distance for candidate filtering are both set to 1.0; the spatial distance multiplier for candidate filtering is 5.0; the metric multiplier for verification is 0.7; the minimum constant inlier ratio for verification is 0.2, which is the same in global loop detection; the minimum ratios of scene overlap for verification in terms of the area and the number of point inliers within the video mask are 0.5 and 0.5, respectively.

Regarding the global loop detection, only keyframes that are at least 10 keyframes away are considered; the multiplier of description similarity for verification is 0.7; the metric multiplier for verification is 0.7; a global loop candidate will be selected if it is at least 10 keyframes away from the ones already selected in a single global loop closure process. In the pose-scale graph optimization for loop closure, the weights of the Relative Pose Scale Factor for non-global and global connections are 1.0 and 5.0, respectively; within this factor, the weights of rotation and scale component, which are ω_{rot} and ω_{scl}, are 5.0 and 0.5, respectively; the weight of the Scale Factor within the loop closure optimization is 10.0; the number of maximum iterations of such optimization is 200; the number of maximum iterations with no relinearization is 5; the relinearization thresholds for pose and scale are 3.0e^{-3} and 1.0e^{-2}.

For the *Mapping* module, settings are as follows. In terms of hyperparameters of factors used in the full factor graph, the weights for the Pose Factor and Scale Factor of the first keyframe are 1.0e^{4}, which are used to anchor the graph in terms of camera pose and depth scale; The Feature-metric Factor and Geometric Consistency Factor

use all samples within the video mask for computation; the Feature-metric Factor has the same weight as the one in camera tracking; the Geometric Consistency Factor has the factor weight of 0.1 and σ_{gc} as 0.03; the weight of the Code Factor is 1.0e^{-4}. In terms of the hyperparameters in factor graph optimization algorithm ISAM2 [250], the relinearization thresholds for camera poses, depth scales, and depth codes are 1.0e^{-3}, 1.0e^{-3}, and 1.0e^{-2}, respectively; partial relinearization check and relinearization skipping are not used; Other parameters in ISAM2 are set to the default ones.

In cases where post-operative processing in a SLAM system is allowed, the *Mapping* and *Loop Closure* modules can be run for an additional amount of time after all frames have been tracked. The *Mapping* module will continue refining the full factor graph. The maximum number of iterations and consecutive no-relinearization iterations are 20 and 5, respectively. In the meantime, the *Loop Closure* module will search for loop pairs for the query keyframes that have not been processed before. When the *Mapping* module finishes, the entire system run will end.

5.5.2 Evaluation Metrics

The metrics used for camera trajectory evaluation are absolute trajectory error (ATE) and relative pose error (RPE) [252]. Note that only the frames that are treated as keyframes by the SLAM system will be evaluated in terms of both trajectory error and depth error. Therefore, synchronization needs to be done to associate the trajec-

tory estimate with the groundtruth one. The trajectory estimate will also be spatially aligned with the pseudo groundtruth trajectory from SfM results in Chapter 2, before computing metrics. The transformation model used for spatial alignment is the similarity transform, and all poses are used to estimate such a transform with the method described in [252].

ATE is used to quantify the whole trajectory and here the form of Root Mean Square Error is used. The rotation and translation components of this metric are defined as

$$\begin{aligned} \mathrm{ATE}_{\mathrm{rot}} &= \left(\frac{1}{N} \sum_{i=0}^{N-1} \left\| \log \left(\boldsymbol{R}_i^{\mathrm{ATE}} \right) \right\|_2^2 \right)^{\frac{1}{2}} \quad \text{and} \\ \mathrm{ATE}_{\mathrm{trans}} &= \left(\frac{1}{N} \sum_{i=0}^{N-1} \left\| \boldsymbol{t}_i^{\mathrm{ATE}} \right\|_2^2 \right)^{\frac{1}{2}} \quad , \end{aligned} \tag{5.15}$$

where $\boldsymbol{R}_i^{\mathrm{ATE}} = \tilde{\boldsymbol{R}}_i^{\mathrm{wld}} \left(\boldsymbol{R}_i^{\mathrm{wld}} \right)^{\intercal}$ and $\boldsymbol{t}_i^{\mathrm{ATE}} = \tilde{\boldsymbol{t}}_i^{\mathrm{wld}} - \boldsymbol{R}_i \boldsymbol{t}_i^{\mathrm{wld}}$. $\tilde{\boldsymbol{R}}_i^{\mathrm{wld}} \in \mathrm{SO}(3)$ and $\tilde{\boldsymbol{t}}_i^{\mathrm{wld}} \in \mathbb{R}^3$ are the groundtruth rotation and translation components of the i^{th} pose in the trajectory, respectively, while $\boldsymbol{R}_i^{\mathrm{wld}} \in \mathrm{SO}(3)$ and $\boldsymbol{t}_i^{\mathrm{wld}} \in \mathbb{R}^3$ are the estimated ones. $N \in \mathbb{R}$ is the number of poses in the synchronized and aligned trajectory estimate.

RPE measures the local accuracy of the trajectory over a fixed frame interval $\Delta \in \mathbb{R}$. This measures the local drift of the trajectory, which is less affected by the loop closure and emphasizes more on the other components of the system. The rotation

and translation components of this metric are defined as

$$\begin{aligned} \mathrm{RPE}_{\mathrm{rot}} &= \left(\frac{1}{N-\Delta} \sum_{i=0}^{N-\Delta-1} \left\| \log\left(\boldsymbol{R}_i^{\mathrm{RPE}} \right) \right\|_2^2 \right)^{\frac{1}{2}} \quad \text{and} \\ \mathrm{RPE}_{\mathrm{trans}} &= \left(\frac{1}{N-\Delta} \sum_{i=0}^{N-\Delta-1} \left\| \boldsymbol{t}_i^{\mathrm{RPE}} \right\|_2^2 \right)^{\frac{1}{2}} \quad , \end{aligned} \tag{5.16}$$

$\boldsymbol{R}_i^{\mathrm{RPE}} \in \mathrm{SO}(3)$ and $\boldsymbol{t}_i^{\mathrm{RPE}} \in \mathbb{R}^3$ are the rotation and translation components of $\boldsymbol{T}_i^{\mathrm{RPE}} \in \mathrm{SE}(3)$, respectively; $\boldsymbol{T}_i^{\mathrm{RPE}}$ is the i^{th} RPE matrix, which is defined as

$$\boldsymbol{T}_i^{\mathrm{RPE}} = \left((\tilde{\boldsymbol{T}}_i^{\mathrm{wld}})^{-1} \tilde{\boldsymbol{T}}_{i+\Delta}^{\mathrm{wld}} \right)^{-1} \left((\boldsymbol{T}_i^{\mathrm{wld}})^{-1} \boldsymbol{T}_{i+\Delta}^{\mathrm{wld}} \right) \quad . \tag{5.17}$$

To evaluate depth estimates, Absolute Relative Difference and Threshold [101] are used. Before computing metrics, different pre-processing is applied for two sets of metrics, which are $\mathrm{ARD}_{\mathrm{traj}}$ and $\mathrm{Threshold}_{\mathrm{traj}}$, and $\mathrm{ARD}_{\mathrm{frame}}$ and $\mathrm{Threshold}_{\mathrm{frame}}$. For the former, the estimated depth per keyframe is re-scaled with the scale component in the similarity transform obtained from the trajectory alignment above. For the latter, the depth estimates are re-scaled so that each estimate has the same scale as the groundtruth one, where the same scaling method in Sec. 3.4.2 is used. In terms

of the definitions of these metrics, ARD is

$$\text{ARD} = \frac{1}{N}\sum_{i=0}^{N-1}\frac{1}{|\Omega_i|}\sum_{\boldsymbol{x}\in\Omega}\frac{|\boldsymbol{D}_i(\boldsymbol{x}) - \tilde{\boldsymbol{D}}_i(\boldsymbol{x})|}{\tilde{\boldsymbol{D}}_i(\boldsymbol{x})} \quad ; \tag{5.18}$$

Threshold is

$$\text{Threshold} = \frac{1}{N}\sum_{i=0}^{N-1}\frac{1}{|\Omega_i|}\sum_{\boldsymbol{x}\in\Omega}\mathbb{1}\left[\max\left(\frac{\boldsymbol{D}_i(\boldsymbol{x})}{\tilde{\boldsymbol{D}}_i(\boldsymbol{x})}, \frac{\tilde{\boldsymbol{D}}_i(\boldsymbol{x})}{\boldsymbol{D}_i(\boldsymbol{x})}\right) < \theta\right] \quad . \tag{5.19}$$

Note that Ω_i here is the region where both scaled depth estimate $\boldsymbol{D}_i \in \mathbb{R}^{1\times H\times W}$ and groundtruth depth $\tilde{\boldsymbol{D}}_i \in \mathbb{R}^{1\times H\times W}$, for the i^{th} synchronized keyframe, have valid depths; $\theta \in \mathbb{R}$ is the threshold used to determine if the depth ratio between the estimate and groundtruth is small enough.

5.5.3 Cross-Subject Evaluation

To evaluate the performance of the SLAM system on endoscopic videos from unseen subjects, we run a cross-validation study. Four models are trained with different train/test splits on the 11 subjects in total. With subjects named as consecutive numbers, the test splits for 4 models are $\{1, 2, 3\}$, $\{4, 5, 6\}$, $\{7, 8, 11\}$, and $\{8, 9, 10\}$, and the train splits for each model are the subjects left. For each subject, several video sequences are available. For evaluation, the proposed SLAM is run on each

testing video and generates estimates of camera poses and dense depth maps for all keyframes. Besides, we also compare against a state-of-the-art feature-based SLAM system, ORB-SLAM3 [54], which we evaluate on all videos at once and use the same set of metrics for evaluation. We adjust the parameters of ORB-SLAM3 so that more keypoint candidates are detected per frame. The evaluation metrics, in Table 5.1, are averaged over all the sequences within the corresponding test split for each of our trained model. Table 5.2 shows the results by averaging each metric over all the sequences for evaluation, where we conduct the paired t-test analysis between the proposed system and ORB-SLAM v3. The results with ***, **, and * stand for p-value smaller than 0.001, 0.01, and 0.05, respectively.

Note that, to make the metrics physically meaningful in terms of the values, we roughly scaled all SfM results before evaluation based on the average size of an adult's nasal cavity. The metric values between methods are not strictly comparable. This is because different sets of images within a sequence are used as keyframes by different methods. However, considering the large number of point samples that are used for computation, the values should approximately indicate the performance difference. Δ in Eq. 5.16 is set to 7 for our results; for ORB-SLAM v3, Δ is set so that the number of original video frames between $\boldsymbol{T}_i^{\mathrm{wld}}$ and $\boldsymbol{T}_{i+\Delta}^{\mathrm{wld}}$ is roughly the same.

Subjects	$\{1,2,3\}$		$\{4,5,6\}$		$\{7,8,11\}$		$\{8,9,10\}$	
Methods / Metrics	Ours	ORB-SLAM v3 [54]	Ours	$\sim$	Ours	$\sim$	Ours	$\sim$
ATE_{trans}(mm)	$\mathbf{1.4 \pm 1.0}$	3.8 ± 2.7	$\mathbf{1.3 \pm 1.7}$	3.8 ± 4.6	$\mathbf{2.2 \pm 1.2}$	6.3 ± 4.8	$\mathbf{1.6 \pm 1.0}$	5.5 ± 3.0
ATE_{rot}(°)	$\mathbf{19.7 \pm 7.8}$	66.2 ± 59.5	$\mathbf{22.8 \pm 17.2}$	61.1 ± 68.1	$\mathbf{25.3 \pm 18.4}$	66.9 ± 48.9	$\mathbf{19.4 \pm 9.5}$	55.8 ± 22.4
RPE_{trans}(mm)	$\mathbf{1.3 \pm 0.4}$	2.5 ± 1.4	$\mathbf{1.4 \pm 0.7}$	2.7 ± 2.1	$\mathbf{1.9 \pm 0.6}$	4.8 ± 3.5	$\mathbf{1.2 \pm 0.5}$	3.6 ± 1.6
RPE_{rot}(°)	$\mathbf{5.9 \pm 1.7}$	6.4 ± 3.5	4.3 ± 2.0	$\mathbf{3.8 \pm 2.6}$	$\mathbf{7.4 \pm 2.6}$	7.7 ± 3.9	$\mathbf{4.5 \pm 1.1}$	8.5 ± 2.9
ARD_{traj}	$\mathbf{0.39 \pm 0.17}$	1.73 ± 1.02	$\mathbf{0.34 \pm 0.10}$	2.00 ± 1.82	$\mathbf{0.38 \pm 0.14}$	1.58 ± 1.42	$\mathbf{0.29 \pm 0.09}$	1.56 ± 1.20
ARD_{frame}	$\mathbf{0.17 \pm 0.04}$	1.73 ± 1.02	$\mathbf{0.17 \pm 0.04}$	2.00 ± 1.82	$\mathbf{0.18 \pm 0.03}$	1.58 ± 1.42	$\mathbf{0.15 \pm 0.02}$	1.56 ± 1.20
$Threshold_{traj}$ ($\theta = 1.25$)	$\mathbf{0.39 \pm 0.19}$	0.15 ± 0.13	$\mathbf{0.46 \pm 0.14}$	0.24 ± 0.21	$\mathbf{0.38 \pm 0.15}$	0.14 ± 0.14	$\mathbf{0.49 \pm 0.13}$	0.14 ± 0.15
$Threshold_{frame}$ ($\theta = 1.25$)	$\mathbf{0.39 \pm 0.19}$	0.15 ± 0.13	$\mathbf{0.46 \pm 0.14}$	0.24 ± 0.21	$\mathbf{0.38 \pm 0.15}$	0.14 ± 0.14	$\mathbf{0.49 \pm 0.13}$	0.14 ± 0.15
$Threshold_{traj}$ ($\theta = 1.25^2$)	$\mathbf{0.70 \pm 0.22}$	0.28 ± 0.22	$\mathbf{0.81 \pm 0.13}$	0.38 ± 0.29	$\mathbf{0.66 \pm 0.16}$	0.27 ± 0.23	$\mathbf{0.84 \pm 0.10}$	0.27 ± 0.22
$Threshold_{frame}$ ($\theta = 1.25^2$)	$\mathbf{0.70 \pm 0.22}$	0.28 ± 0.22	$\mathbf{0.81 \pm 0.13}$	0.38 ± 0.29	$\mathbf{0.66 \pm 0.16}$	0.27 ± 0.23	$\mathbf{0.84 \pm 0.10}$	0.27 ± 0.22

Table 5.1: **Cross-subject evaluation on SLAM systems per test split.** Note that $\sim$ is used as the name abbreviation of the comparison method.

Metrics / Methods	ATE_{trans} (mm)	ATE_{rot} (°)	RPE_{trans} (mm)	RPE_{rot} (°)	ARD_{traj}	ARD_{frame}	$Threshold_{traj}$ ($\theta = 1.25$)	$Threshold_{frame}$ ($\theta = 1.25$)	$Threshold_{traj}$ ($\theta = 1.25^2$)	$Threshold_{frame}$ ($\theta = 1.25^2$)
Ours	$\mathbf{1.6 \pm 1.4}$	$\mathbf{22.2 \pm 15.1}$	$\mathbf{1.5 \pm 0.6}$	$\mathbf{5.5 \pm 2.4}$	$\mathbf{0.36 \pm 0.16}$	$\mathbf{0.17 \pm 0.03}$	$\mathbf{0.42 \pm 0.17}$	$\mathbf{0.73 \pm 0.08}$	$\mathbf{0.74 \pm 0.21}$	$\mathbf{0.95 \pm 0.04}$
ORB-SLAM v3 [54]	$4.7 \pm 4.2^{***}$	$62.5 \pm 55.5^{***}$	$3.5 \pm 2.5^{***}$	6.3 ± 3.6	$1.76 \pm 1.49^{***}$	$24.27 \pm 42.07^{**}$	$0.17 \pm 0.18^{***}$	$0.37 \pm 0.13^{***}$	$0.31 \pm 0.25^{***}$	$0.56 \pm 0.15^{***}$

Table 5.2: **Cross-subject evaluation on SLAM systems.**

FT	FM	RT	Local	Global	$\mathrm{ATE}_{\mathrm{trans}}(\mathrm{mm})$	$\mathrm{ATE}_{\mathrm{rot}}(°)$	$\mathrm{RPE}_{\mathrm{trans}}(\mathrm{mm})$	$\mathrm{RPE}_{\mathrm{rot}}(°)$
✓	✓	✓	✓	✓	1.6 ± 1.4	22.2 ± 15.1	1.5 ± 0.6	5.5 ± 2.4
	✓	✓	✓	✓	$3.4 \pm 2.7^{***}$	$43.3 \pm 27.9^{***}$	$2.6 \pm 1.4^{***}$	$7.3 \pm 3.0^{***}$
		✓	✓	✓	$3.3 \pm 2.8^{***}$	$40.2 \pm 23.6^{***}$	$2.6 \pm 1.2^{***}$	$7.0 \pm 2.6^{***}$
✓	✓		✓	✓	2.7 ± 5.5	23.8 ± 14.5	2.1 ± 3.2	5.3 ± 2.1
✓	✓	✓	✓		$2.0 \pm 1.9^{*}$	$26.8 \pm 21.2^{*}$	1.5 ± 0.7	5.5 ± 2.4
✓	✓	✓			$2.0 \pm 1.9^{*}$	$25.5 \pm 18.5^{*}$	1.5 ± 0.7	5.4 ± 2.4

Table 5.3: **Ablation study for the SLAM system on trajectory-related metrics.** FT, FM, RT, Local, Global stand for the Feature-metric Factor in the *Camera Tracking* module, the Feature-metric Factor in the *Mapping* module, the Reprojection Factor in the *Camera Tracking* module, local loop detection in the *Loop Closure* module, and global loop detection and closure in the *Loop Closure* module, respectively. We conduct the paired t-test analysis for results of all the sequences between an ablation run and the standard run shown in the first row of this table. The results with ***, **, and * stand for p-value smaller than 0.001, 0.01, and 0.05, respectively. As can be seen, the Feature-metric Factor has a large impact on both trajectory and trajectory-scaled depth metrics; the Reprojection Factor mainly affects trajectory metrics; the *Loop Closure* module mainly affects the trajectory metrics $\mathrm{ATE}_{\mathrm{trans}}$ and $\mathrm{ATE}_{\mathrm{rot}}$.

5.5.4 Ablation Study

We evaluate the contributions of several components in the SLAM system by disabling some components in different runs. The components for ablation are the Feature-metric Factor in the *Camera Tracking* and *Mapping* modules, Reprojection Factor in the *Camera Tracking* module, local loop detection in the *Loop Closure* module, and global loop detection and closure in the *Loop Closure* module. All metrics described in Sec. 5.5.3 are evaluated in this ablation study. The results are shown in Table 5.3 and 5.4. Note that the value of each metric is averaged over all the sequences from all subjects, where each subset of the sequences is evaluated with the corresponding trained model so that all the sequences are unseen during training.

FT	FM	RT	Local	Global	ARD_{traj}	$\text{ARD}_{\text{frame}}$	$\text{Threshold}_{\text{traj}}$ ($\theta = 1.25$)	$\text{Threshold}_{\text{frame}}$ ($\theta = 1.25$)	$\text{Threshold}_{\text{traj}}$ ($\theta = 1.25^2$)	$\text{Threshold}_{\text{frame}}$ ($\theta = 1.25^2$)
✓	✓	✓	✓	✓	0.36 ± 0.16	0.17 ± 0.03	0.42 ± 0.17	0.73 ± 0.08	0.74 ± 0.21	0.95 ± 0.04
	✓	✓	✓	✓	$0.49 \pm 0.19^{***}$	0.17 ± 0.03	$0.29 \pm 0.16^{***}$	0.73 ± 0.08	$0.59 \pm 0.23^{***}$	0.95 ± 0.04
		✓	✓	✓	$0.50 \pm 0.25^{**}$	0.17 ± 0.03	$0.32 \pm 0.17^{**}$	0.74 ± 0.08	$0.61 \pm 0.24^{**}$	0.95 ± 0.04
✓	✓		✓	✓	0.35 ± 0.15	0.17 ± 0.03	0.43 ± 0.17	0.73 ± 0.08	0.76 ± 0.18	0.95 ± 0.04
✓	✓	✓	✓		0.36 ± 0.16	0.17 ± 0.03	0.42 ± 0.17	0.73 ± 0.08	0.74 ± 0.21	0.95 ± 0.04
✓	✓	✓			$0.35 \pm 0.16^{*}$	0.17 ± 0.03	0.42 ± 0.18	0.73 ± 0.08	0.74 ± 0.22	0.95 ± 0.04

Table 5.4: **Ablation study for the SLAM system on depth-related metrics.** The settings and notations are the same as Table 5.3.

5.5.5 Evaluation with CT

This study uses the residual error metric described in Sec. 3.6.2. Before computing the residual error, several pre-processing steps are required. First, the method described in Sec. 3.5.2 and 3.5.3 is applied to obtain a surface reconstruction from the depth maps and camera poses estimated by the proposed SLAM system. The slope in the truncated signed distance function is constant instead of the depth uncertainty which is used in Sec. 3.5.2. Then a point cloud registration algorithm based on [120] is applied between the surface reconstruction and the CT surface model, where a similarity transform is estimated. Note that before the registration, a manual initial alignment between these two models is applied. After the registration finishes, the residual error is computed between the registered surface reconstruction and the CT surface model.

In this study, we evaluate the accuracy of surface reconstructions from the videos of the four cadavers, where for each subject, the metrics of all the sequences are

averaged over to report here. The average residual errors for subject 7, 9, 10, and 11 are 0.83, 0.88, 0.78, and 0.86 mm, respectively.

5.6 Discussion

The accuracy of trajectory estimation depends on how consistent and distinctive the feature and descriptor maps are, as well as the accuracy of the depth estimates. Though the depth is optimizable during the SLAM running, the depth basis maps estimated from the input image still bound the variation mode of the final depth estimate. Therefore, if the depth network is not familiar with the scene, it is probable that a depth estimate close to the truth will not be obtained. Therefore, a representative collection of training data is crucial for the generalizability of such a learning-based SLAM system.

For the current system that is trained on sinus endoscopy dataset, we would expect the system generalizes decently to endoscopy on tubular structures, such as bronchoscopy. It could be less generalizable to domain such as laparoscopy because the overall geometry of the anatomy is unseen for the depth network. However, we would still expect the appearance representation to generalize well to these more distant cases because it only models texture instead of geometry. It is also expected the system can generalize even better if the capacity of the network is configured to be larger

with a larger dataset for training. The networks do not produce uncertainty estimates for now, and those could potentially further improve the generalizability and benefit the factor graph optimization if being accurate.

Currently, the system cannot recover from a spurious global loop connection and therefore the global loop detection criteria need to be strict to keep the false positive rate to zero. Such an error could potentially be detected by monitoring the overall objective of the full factor graph after each global loop closure [223]. For now, camera relocalization after tracking failure is not implemented and is likely required in cases where images with bad conditions (*e.g.*, image blurring) happen often, such as laparoscopy; a method similar to the searching in the global loop detection could be used. A keyframe culling method can be implemented to reduce the number of keyframes to reduce the memory requirement and accelerate the computation. A method similar to the reference keyframe selection in the Camera Tracking module may be used to find redundant keyframes.

The proposed SLAM system is currently designed for static scenes. Nevertheless, having additional optimization variables per keyframe to model geometry deformation could potentially make the system suitable for a deformable environment. As for the robustness to a dynamic environment with changing textures (*e.g.*, changing illumination and bleeding), it depends on whether there are similar conditions in the training dataset and how large the affected regions are within images. For example,

if the bleeding region is small relative to the entire image, even if the blood moves during the video capturing, the impact should be minimal because factors, such as Feature-metric Factor, will focus on the large unaffected regions during optimization.

5.7 Conclusion

In this chapter, we propose a SLAM system that is robust to texture-scarce scenarios with learning-based appearance and geometric representations. An effective training scheme is developed to learn such representations that are suitable for the SLAM system run. In the experiments, we show that the proposed system performs favorably compared with a state-of-the-art feature-based SLAM system in terms of the accuracy of both camera trajectories and geometry estimates.

The proposed SLAM system currently only works in the static environment. However, it is feasible to add another type of optimization variable to factor graph optimization to take care of the tissue deformation, such as the deformation-spline used in [253], and therefore worthy of working on as a future direction. Similar to the directions in Chapter 2, it is also worth exploring how to make such a SLAM system work in scenarios where the topology of anatomy is changed due to surgical operations. An additional map for each keyframe to notate which part of the region is unaffected could be one way to achieve this. Currently, the global loop closure needs

to have a zero false-positive rate to have reasonable performance, it is desired to have a failure-aware and recovery mechanism to relax such a constraint. Based on our observation, having enough global loop closures is critical for accurate estimation of the camera trajectory, a study on what scoping path is the best for each type of endoscopy could be worth exploring. Currently, for the robustness of the system, the local connection only considers the spatially closest pairs because a keyframe connection with a small scene overlap could be erroneous. Having accurate mid-range connections could further improve the performance and reduce the drifting errors even when no global loops are available, which was observed in [54].

For the task of surface reconstruction and endoscope tracking from a video with pre-operative model alignment, with the works developed in this thesis, there are in general one retrospective pipeline and one online one. The retrospective pipeline can already be fully built with the works described in this thesis, as described in Sec. 1.2.3. For the online pipeline described in this chapter, however, if an automatic alignment between the pre-operative and intra-operative surface model with iterative refinement is needed, some additional works are required. The registration method developed in Chapter 4 considers one-time model alignment and did not exploit the fact that a real-time SLAM system updates the map whenever a new scene is observed. Integrating such surface update into the optimization step of a registration method could potentially further improve the registration performance in terms of ac-

curacy and processing speed. Besides, to align the pre-operative and intra-operative models during a SLAM system run, a single surface model for the entire observed environment needs to be built and updated. In this work, the dense depth estimates are not fused during the system run and a real-time depth fusion and surface extraction method needs to be developed to obtain such a surface model.

Chapter 6

Summary and Future Work

6.1 Summary

In this thesis, we have described two pipelines, one being retrospective and the other being real-time, for surface reconstruction from a monocular endoscopic video. We try to combine the strength of the expressivity of deep learning approaches and the rigorousness and accuracy of the traditional non-linear optimization to tackle some of the challenges present in vision-based methods for endoscopy, such as texture scarceness, changing illumination, and multimodality, leading to better performance in terms of robustness and accuracy.

To summarize the contributions of the thesis:

- In Chapter 2, we propose a retrospective sparse reconstruction algorithm that

can estimate point cloud with high accuracy and density and camera trajectory with more completeness from a monocular video. To enable this, we develop a deep learning-based dense image feature descriptor that can establish dense and accurate point correspondences between video frames, which is applied to the pair-wise feature matching stage of a standard SfM pipeline.

- In Chapter 3, we describe a retrospective surface reconstruction pipeline that can estimate a dense surface model from a monocular video. The pipeline is patient-specific with self-supervised depth and descriptor learning involved, which enables the method to take advantage of the high expressivity of deep learning and avoids the need to generalize to unseen subjects. The traditional multi-depth fusion and surface extraction methods are used to ensure that the estimates from the first part of the pipeline can be merged effectively.

- In Chapter 4, we introduce a global registration algorithm for point cloud data that is robust to resolution mismatch that often happens in the multi-modal scenario. Specifically, we develop a network normalization technique that is shown to help a 3D network produce more consistent and distinctive geometric features for samples with different resolutions. These geometric features can establish more accurate point correspondences between samples and enable the application of an optimization-based global point cloud registration method to

align two models in scenarios such as video-CT registration.

- In Chapter 5, we design a real-time SLAM system that can estimate a surface geometry and camera trajectory from a monocular video. We exploit the deep learning-based representation in terms of both geometry and appearance, together with a non-linear factor graph optimization, to enable such a system.

6.2 Future Directions

With the works in this thesis, several clinical studies can be conducted as future works. With the trajectory estimation in Chapter 2, endoscope trajectories from a large number of endoscopic videos can be obtained. These data are valuable for large-scale trajectory analysis, which may find specific patterns, such as the difference between expert and novice endoscopists, with valuable insights from such analysis. The surface models obtained with the pipeline in Chapter 3 are decently accurate and therefore can be used for clinic-related measurements (*e.g.*, cross-sectional area of nasal cavity). Because endoscopy inspection can be performed frequently in an outpatient setting and therefore it enables longitudinal analysis of certain treatments on a large scale. For example, one study to conduct is to quantitatively analyze the effectiveness of nasal polyp shrinking treatment through longitudinal volume measuring of polyps. As a decent percentage of regions (*e.g.*,23% for colonoscopy in Hong *et*

al. [8]) can often be missed during endoscopy inspection, it will be valuable if a clinical study on large-scale missing region analysis for endoscopy inspection can be conducted. These may potentially help to find some patterns to provide insights, such as which regions are mostly missed.

Besides the research directions related to this thesis, many other valuable ones exist that can potentially enable more endoscopic applications. For the SLAM system, with the development for real-time registration mentioned above, information from pre-operative volume data, such as critical underlying structures, can be overlaid onto the endoscopic images. To display these structures in an augmented reality manner, a 3D segmentation needs to be developed to segment structures (*e.g.*, facial nerve) underneath the observable anatomy from the endoscope. In this thesis, we assume there are no instrument movements in endoscopic videos. With 2D instrument segmentation, the moving region of the images can be ignored for the pipelines of this thesis and the applicability of these works can thus be extended. 3D instrument tracking can track the pose of the instrument that appears in the endoscopic video. This enables collecting instrument movement information from endoscopic videos, which provides guidance of surgical operations from experts. Together with the trajectory estimates from Chapter 2 and the surface model from Chapter 3, these are very valuable in providing expert guidance for endoscopy training simulators. By combining the 3D instrument tracking with the SLAM system developed in Chapter 5, an intelligent

endoscope holder could be developed to help surgeons hold and move the endoscope during operations.

Appendix A

Supplementary Material for Chapter 4

A.1 Transposed NHN-Conv and B-NHN-Conv

Since it is not straightforward to describe transposed sparse 3D convolution in mathematical terms, we described it in words here instead. For equations in Sec. 4.3, $\sum_{v \in \mathcal{N}(u)}$ indicates a generalized sparse convolution operation in actual

Materials in this appendix are from Liu *et al*. [18].

implementation. We used the *MinkowskiConvolutionFunction* in the python package *Minkowski Engine* [163] for this purpose. To implement a transposed version of the NHN-Conv and B-NHN-Conv, we simply replaced all *MinkowskiConvolutionFunction* with *MinkowskiConvolutionTransposeFunction*.

A.2 Architectures of comparison methods

MinkowskiNet with standalone normalization. The architecture is shown in Fig. A.1. In Sec. 4.6, we evaluated this architecture with normalization BatchNorm [149], InstanceNorm [152], and Batch-Instance Norm [156]. These are abbreviated as Mink.+BN, Mink.+IN, and Mink.+BIN. Mink.+BN, Mink.+IN, and Mink.+BIN all have around 8.80 million learnable parameters. In addition, Mink.+NHN and Mink.+B-NHN also have around 8.80 million learnable parameters.

FCGF. The architecture is shown in Fig. A.2. Please find the mathematical definition of the 3DConv layer in Sec. 4.3. Tr-3DConv is simply a transposed version of 3DConv. All numbers mean the same as the ones in Sec. 4.4. For 3DConv and Tr-3DConv, the three numbers mean kernel size along one spatial dimension, stride size, and output channel size. The number in BatchNorm and ResBlock represents the output channel size. The total number of learnable parameters is 8.76 million.

KPConv. The architecture is shown in Fig. A.3. We changed the hyperparameter

APPENDIX A

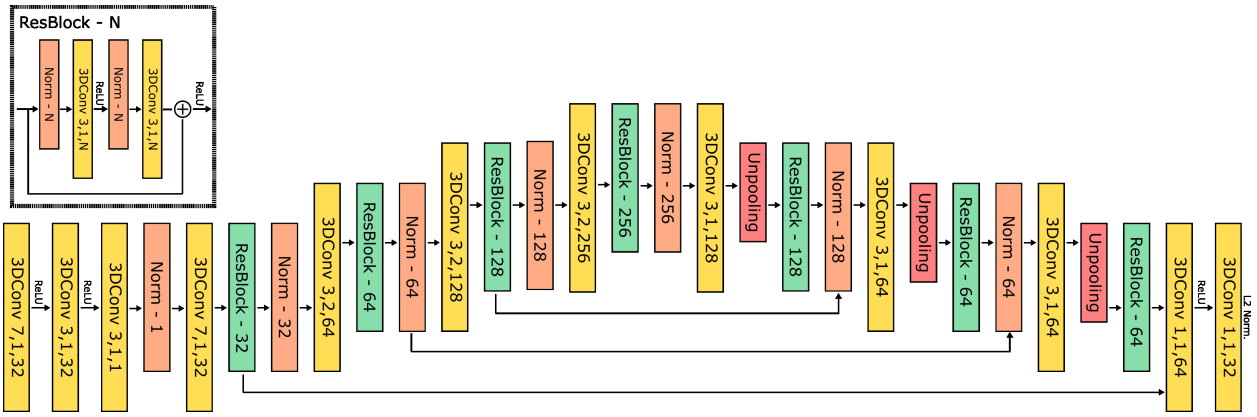

Figure A.1: **Network architecture for MinkowskiNet with standalone normalization.** Norm can be any choice of standalone normalization. The number after Norm is the output channel size of this module. For the transposed version of this architecture that was used in the standard 3DMatch benchmark, we simply replaced the combination of the 3DConv and Unpooling with a transposed 3DConv with stride size 2. Note all skipping connections in this section are concatenation along channel dimension.

setting in the original work [169] for the 3DMatch dataset and the task of 3D descriptor learning. First, all the parameters are kept the same. The number of kernel points per filter is 15. The first subsampling grid size is set to 5 cm for a fair comparison with other methods in the 3DMatch benchmark. The first radius, *i.e.* number of grid cells, of convolution is 2.5. The radius of the area under influence for each kernel point is 1.2 grid cells. The type of KPConv influence is linear. The aggregation mode is summation in the standard benchmark and averaging in the resolution-mismatch one. The centered 3D spatial locations of all points are used for neighbor searching and downsampling inside the architecture. For what is changed, the channel size of the input feature is 1. The input features are all one. The channel dimension of the

APPENDIX A

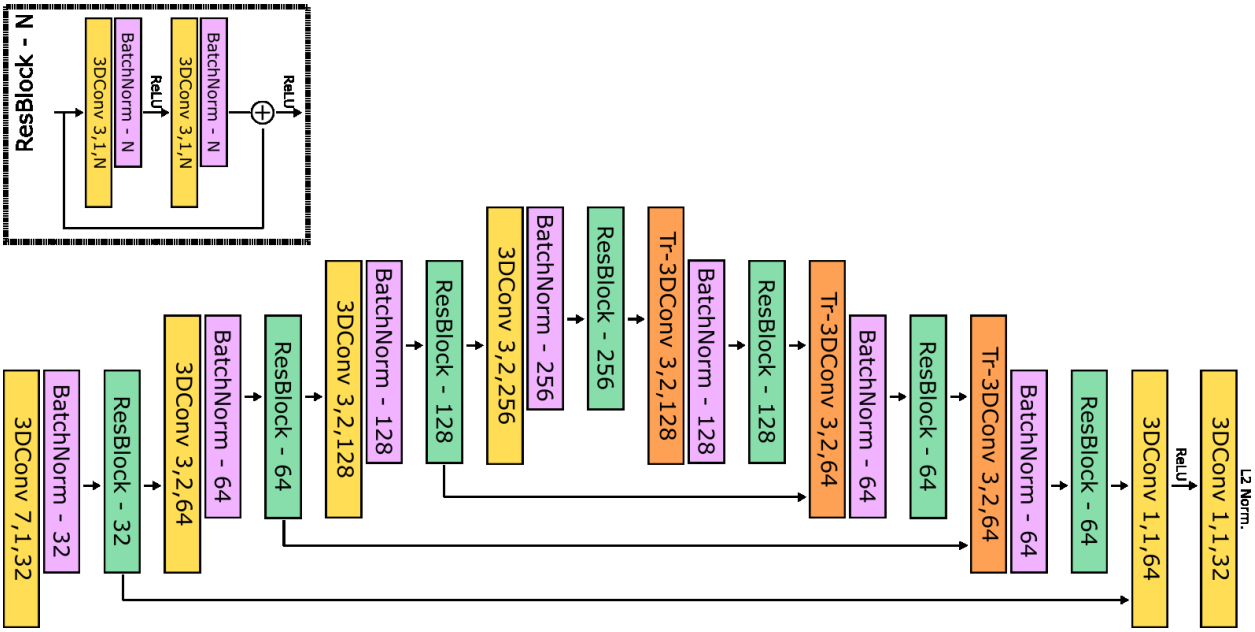

Figure A.2: **Network architecture for FCGF [59].** Note that the architecture used in the actual state-of-the-art model in [59] is different from the one they have in the paper. © 2021 IEEE

APPENDIX A

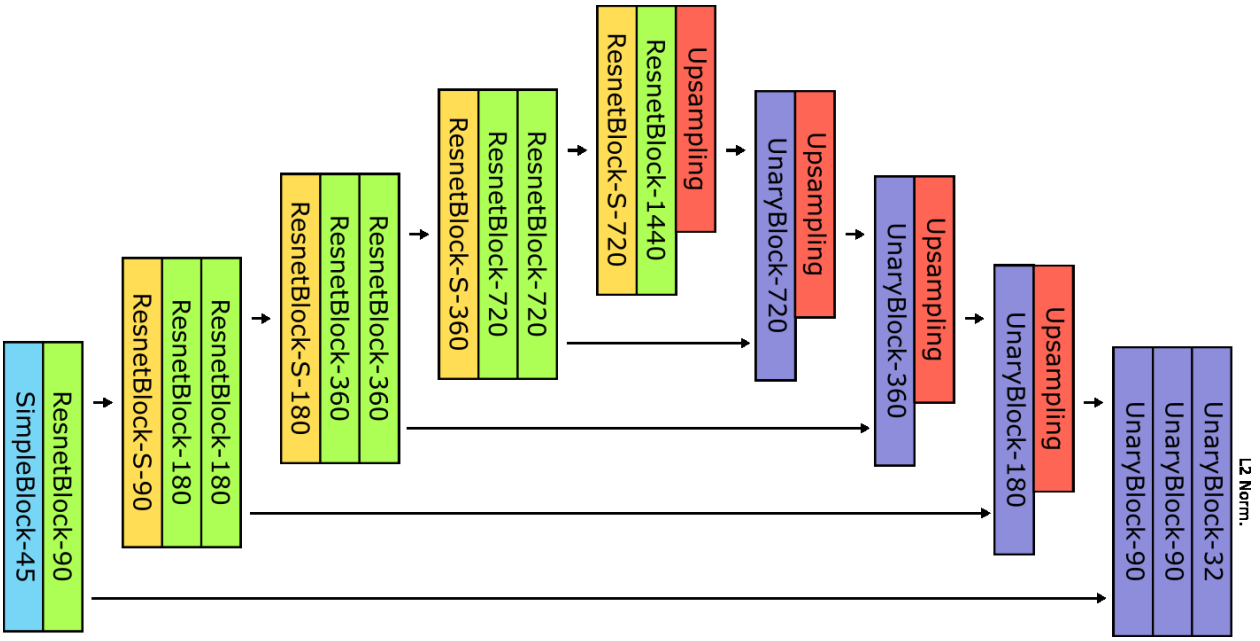

Figure A.3: **Network architecture for KPConv [169].** Note we modified the original architecture for the task of 3D descriptor learning. The number besides the module name is the size of the output channel. Please refer to github repo `https://github.com/HuguesTHOMAS/KPConv-PyTorch` for the implementation details of all the modules in the figure. In the figure, SimpleBlock stands for the *SimpleBlock* module; ResnetBlock stands for the *ResnetBottleneckBlock* module; ResnetBlock-S stands for the *ResnetBottleneckBlock* module with striding enabled; Upsampling stands for the *NearestUpsampleBlock* module; UnaryBlock stands for the *UnaryBlock* module. © 2021 IEEE

filter base is 90. The training setting, such as batch size, optimizer, and loss function, etc, is the same as the Mink. architecture described in Sec. 4.4. The total number of learnable parameters is 9.08 million.

PPNet. The architecture is shown in Fig. A.4. Some hyperparameter settings and the architecture is changed, compared with the original work [220]. The channel dimension of the input feature is 3, which are all constant number one. The input

APPENDIX A

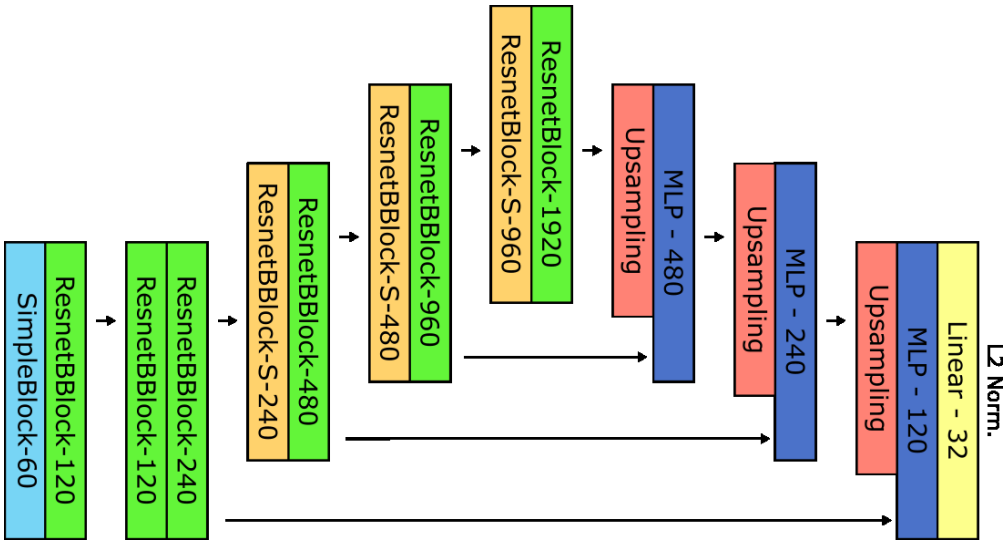

Figure A.4: **Network architecture for PPNet [220].** Note we modified the original architecture for the task of 3D descriptor learning. The number besides the module name is the size of the output channel. We use the provided PPNet modules in the *PyTorch Points 3D* python package. In the figure, SimpleBlock stands for the *SimpleBlock* module; ResnetBBlock stands for the *ResnetBBlock* module; ResnetBBlock-S stands for the *ResnetBBlock* module with striding enabled. The combination of Upsamling and MLP modules stands for the *FPModule_PD* module. The Linear module at the end is a simple linear transform. © 2021 IEEE

grid size is set to 5 cm for a fair comparison. After each downsampling layer, the grid size is multiplied by 2. The channel dimension of the filter base is 60. The maximum number of neighbors is set to 27. The position embedding type is ”XYZ” and the reduction type for local aggregation is averaging. The upsampling modules are the nearest upsampling. The total number of learnable parameters is 9.07 million.

PointNet++. The architecture is shown in Fig. A.5. The input vertex features are a concatenation of centered point XYZ location and constant one. In MSGD, as opposed to the original design where the point cloud is downsampled to a fixed num-

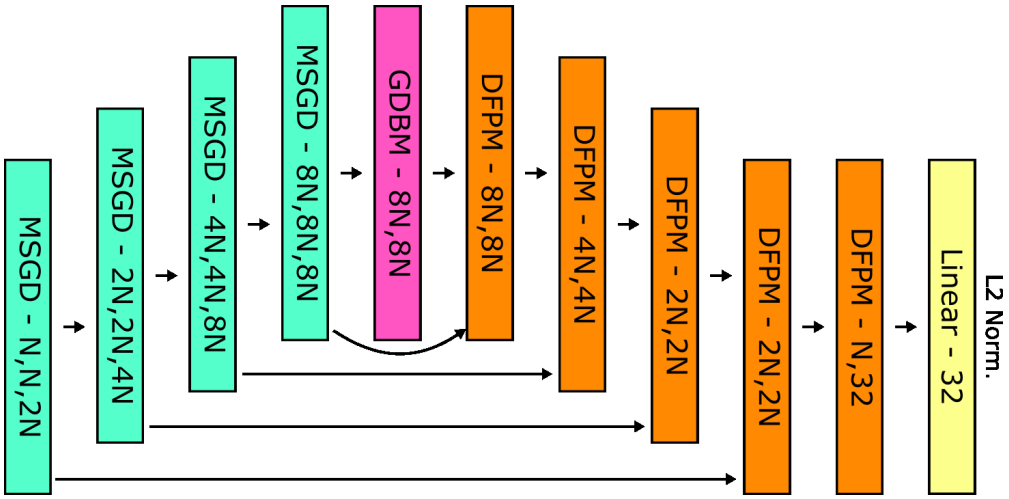

Figure A.5: **Network architecture for PointNet++ [165]** Note we modified the original architecture for the task of 3D descriptor learning. We use the provided PointNet2 modules in the *PyTorch Points 3D* python package. In the figure, MSGD stands for the *PointNetMSGDown* module; GDBM stands for the *GlobalDenseBaseModule* module; DFPM stands for the *DenseFPModule* module. The Linear module at the end is a simple linear transform. MSGD consists of point downsampling and three Linear layers, and the three numbers after the module name are the output channel sizes of these Linear layers. The two numbers after GDBM and DFPM are the output channel sizes of the two Linear layers within the module. The number after Linear is the output channel size of the module. N, as the filter base, is set to 112. © 2021 IEEE

ber, we use a fixed ratio of the points instead to account for the varying sample size. The downsample ratios for the four MSGD modules are 1.0, 0.25, 0.25, and 0.25. For all MSGD and GDBM modules, an additional 3 channels of point locations are concatenated with the feature map. The maximum number of neighbors is set to 27. The initial neighborhood radius is 12.5 cm. The radius is multiplied by 2 or divided by 2 whenever the point cloud is downsampled or upsampled, respectively. The total number of learnable parameters is 8.93 million.

DCM-Net. Because the form of input data in the experiments is a point cloud, the architecture in the original work [176] that uses only K-nearest neighbors for message propagation is used. The input vertex features point locations. The filters of the encoder part are [16, 96, 256, 384] with the number of propagation steps per graph layer as 4. The filters of the decoder part are the same as the encoder part, which is the original design in [176]. The pooling and aggregation modes are set to ”max” and ”mean”, respectively. The channel size of the output feature description is 32, the same as all other comparison methods. The total number of learnable parameters is 7.29 million.

A.3 Visualization of feature embeddings

The output feature embeddings from Mink.+B-NHN are visualized in Fig. A.6, Fig. A.7, and Fig. A.8 for the clinical datasets, the 3DMatch [136], and the KITTI odometry [148], respectively. The models of Mink.+B-NHN are trained with the resolution-mismatch settings described in Sec. 4.6. UMAP [221] is used to reduce 32-dimension output feature descriptions to scalar values. These are then displayed with the JET colormap. To better visualize the embeddings of the 3DMatch and clinical datasets, we display the meshes instead of the input point clouds. The vertices of a displayed mesh get the embeddings of the spatially closest point in the correspond-

ing input point cloud. All sample pairs displayed in these figures have a resolution mismatch. The mesh edges of the samples from 3DMatch and clinical datasets are displayed to make the resolution mismatch easier to observe. If the displayed colors of feature embeddings are similar, the L2 distances between the original feature embeddings are probably small.

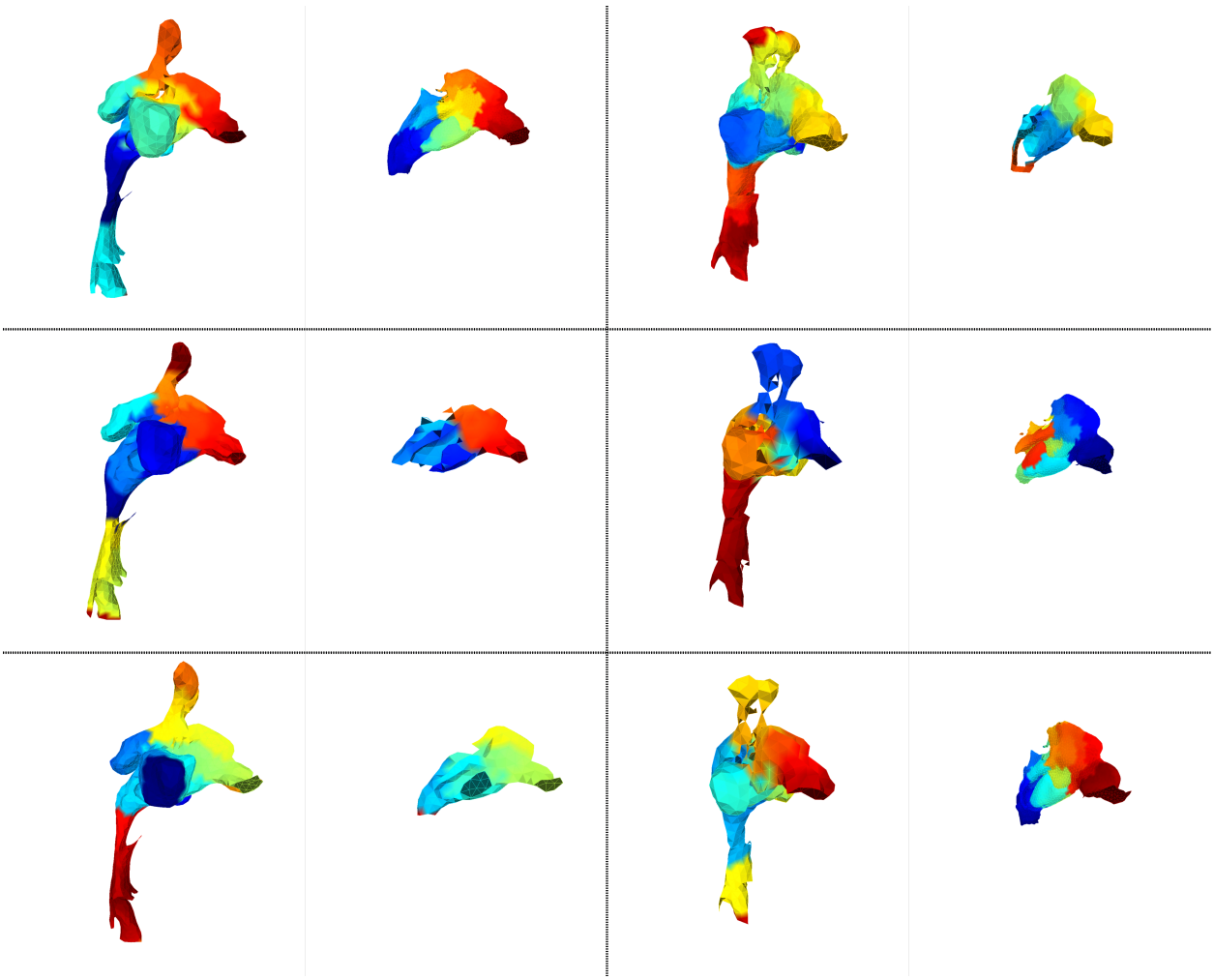

Figure A.6: **Visualization of feature embeddings for the clinical dataset of nasal cavities.** Matching colors indicate closely aligned feature representations. The 1st and 2nd columns form sample pairs, the same for the 3rd and 4th columns. The 1st and 3rd columns display the entire nasal cavity, while the 2nd and 4th columns display the nasal passage.

APPENDIX A

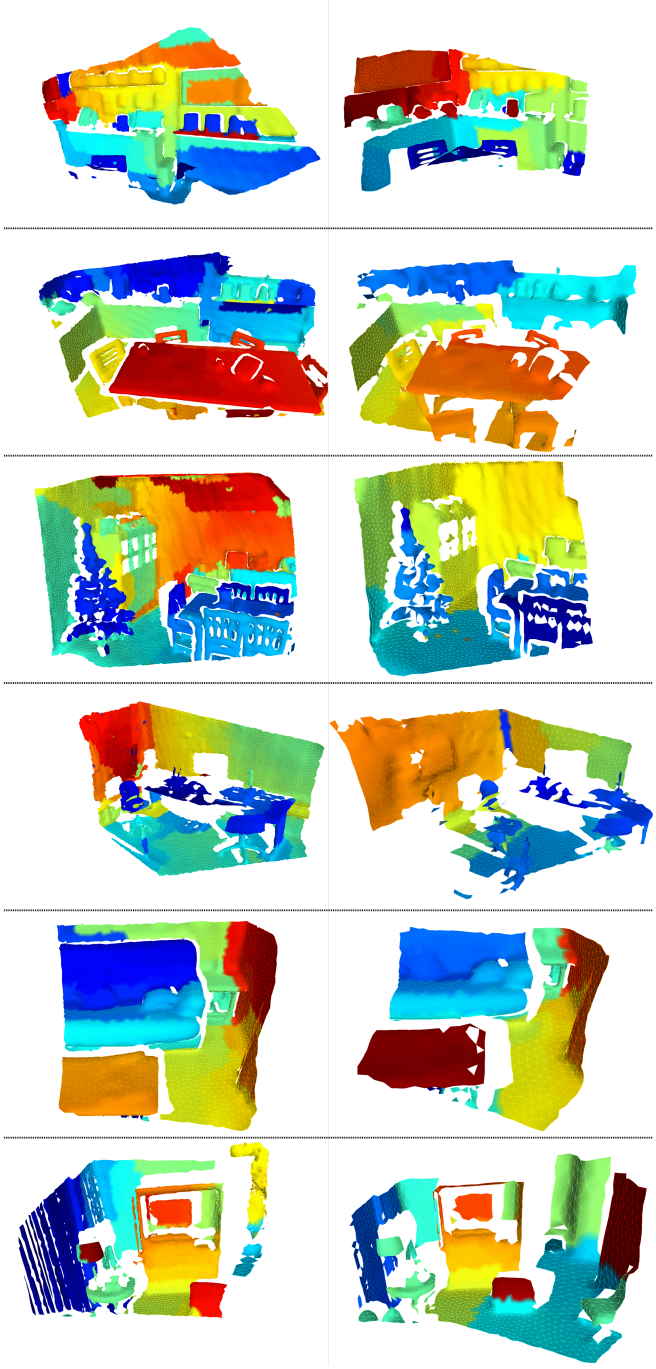

Figure A.7: **Visualization of feature embeddings for the 3DMatch dataset [136].** Matching colors indicate closely aligned feature representations. The 1st and 2nd columns form sample pairs. © 2021 IEEE

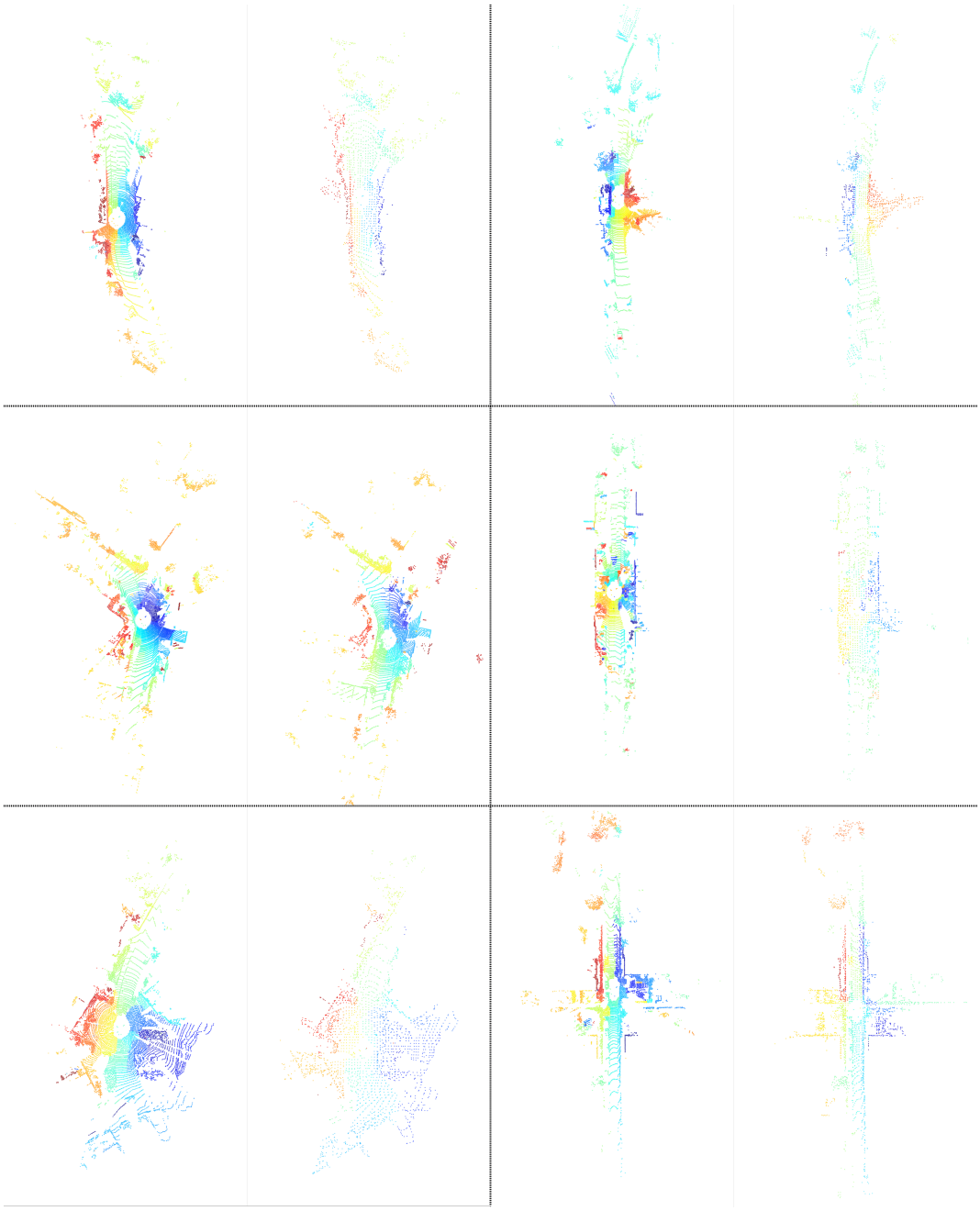

Figure A.8: **Visualization of feature embeddings for the KITTI dataset [148].** Matching colors indicate closely aligned feature representations. The 1st and 2nd columns form sample pairs, the same with the 3rd and 4th columns.

Bibliography

[1] P. C. De Groen, "History of the endoscope [scanning our past]," *Proceedings of the IEEE*, vol. 105, no. 10, pp. 1987–1995, 2017.

[2] C. Nezhat, "History of endoscopy," http://sls.org/nezhats-history-of-endoscopy/, 2005.

[3] K. Ball, *Endoscopic Surgery*, ser. Mosby's perioperative nursing series. Mosby, 1997.

[4] C. J. Powers, "A brief history of endoscopy," *Seminars in Perioperative Nursing*, vol. 2, no. 3, pp. 129–132, Jul. 1993.

[5] B. I. Hirschowitz, "A personal history of the fiberscope," *Gastroenterology*, vol. 76, no. 4, pp. 864–869, Apr. 1979.

[6] M. Finocchiaro, P. Cortegoso Valdivia, A. Hernansanz, N. Marino, D. Amram, A. Casals, A. Menciassi, W. Marlicz, G. Ciuti, and A. Koulaouzidis, "Training

simulators for gastrointestinal endoscopy: Current and future perspectives," *Cancers*, vol. 13, no. 6, Mar. 2021.

[7] L. X. Harrington, J. W. Wei, A. A. Suriawinata, T. A. Mackenzie, and S. Hassanpour, "Predicting colorectal polyp recurrence using time-to-event analysis of medical records," *AMIA Jt Summits Transl Sci Proc*, vol. 2020, pp. 211–220, May 2020.

[8] W. Hong, J. Wang, F. Qiu, A. Kaufman, and J. Anderson, "Colonoscopy simulation," in *Medical Imaging 2007: Physiology, Function, and Structure from Medical Images*, vol. 6511. International Society for Optics and Photonics, 2007, p. 65110R.

[9] P. Fockens, "Endoscopic management of perforations in the gastrointestinal tract," *Gastroenterol. Hepatol.*, vol. 12, no. 10, pp. 641–643, Oct. 2016.

[10] R. Eliashar, J.-Y. Sichel, M. Gross, E. Hocwald, I. Dano, A. Biron, A. Ben-Yaacov, A. Goldfarb, and J. Elidan, "Image guided navigation system-a new technology for complex endoscopic endonasal surgery," *Postgrad. Med. J.*, vol. 79, no. 938, pp. 686–690, Dec. 2003.

[11] M. Hytönen, K. Blomgren, M. Lilja, and A. Mäkitie, "How we do it: septoplasties under local anaesthetic are suitable for short stay surgery; the clinical outcomes." *Clinical otolaryngology: official journal of ENT-UK; official journal*

of Netherlands Society for Oto-Rhino-Laryngology & Cervico-Facial Surgery, vol. 31, no. 1, pp. 64–68, 2006.

[12] T. F. P. Bezerra, M. G. Stewart, M. A. Fornazieri, R. R. de Mendonca Pilan, F. de Rezende Pinna, F. G. de Melo Padua, and R. L. Voegels, "Quality of life assessment septoplasty in patients with nasal obstruction," *Brazilian journal of otorhinolaryngology*, vol. 78, no. 3, pp. 57–62, 2012.

[13] A. L. Feng, C. R. Razavi, P. Lakshminarayanan, Z. Ashai, K. Olds, M. Balicki, Z. Gooi, A. T. Day, R. H. Taylor, and J. D. Richmon, "The robotic ent microsurgery system: a novel robotic platform for microvascular surgery," *The Laryngoscope*, vol. 127, no. 11, pp. 2495–2500, 2017.

[14] J. J. McGoran, M. E. McAlindon, P. G. Iyer, E. J. Seibel, R. Haidry, L. B. Lovat, and S. S. Sami, "Miniature gastrointestinal endoscopy: Now and the future," *World J. Gastroenterol.*, vol. 25, no. 30, pp. 4051–4060, Aug. 2019.

[15] X. Liu, Y. Zheng, B. Killeen, M. Ishii, G. D. Hager, R. H. Taylor, and M. Unberath, "Extremely dense point correspondences using a learned feature descriptor," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 4847–4856.

[16] X. Liu, A. Sinha, M. Ishii, G. D. Hager, A. Reiter, R. H. Taylor, and M. Unberath,

"Dense depth estimation in monocular endoscopy with self-supervised learning methods," *IEEE transactions on medical imaging*, 2019.

[17] X. Liu, M. Stiber, J. Huang, M. Ishii, G. D. Hager, R. H. Taylor, and M. Unberath, "Reconstructing sinus anatomy from endoscopic video – towards a Radiation-Free approach for quantitative longitudinal assessment," in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2020*. Springer International Publishing, 2020, pp. 3–13.

[18] X. Liu, B. D. Killeen, A. Sinha, M. Ishii, G. D. Hager, R. H. Taylor, and M. Unberath, "Neighborhood normalization for robust geometric feature learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021, pp. 13 049–13 058.

[19] J. L. Schönberger and J. Frahm, "Structure-from-motion revisited," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, pp. 4104–4113.

[20] R. Roberts, S. N. Sinha, R. Szeliski, and D. Steedly, "Structure from motion for scenes with large duplicate structures," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 3137–3144, 2011.

[21] N. Jiang, P. Tan, and L. F. Cheong, "Seeing double without confusion: Structure-

from-motion in highly ambiguous scenes," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1458–1465, 2012.

[22] C. Sweeney, T. Sattler, T. Hollerer, M. Turk, and M. Pollefeys, "Optimizing the viewing graph for structure-from-motion," *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2015 Inter, pp. 801–809, 2015.

[23] C. Kong and S. Lucey, "Deep non-rigid structure from motion," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 1558–1567.

[24] C. Wang, C.-H. Lin, and S. Lucey, "Deep NRSfM++: Towards unsupervised 2D-3D lifting in the wild," in *2020 International Conference on 3D Vision (3DV)*, Nov. 2020, pp. 12–22.

[25] K. L. Lurie, R. Angst, D. V. Zlatev, J. C. Liao, and A. K. Ellerbee Bowden, "3D reconstruction of cystoscopy videos for comprehensive bladder records," *Biomed. Opt. Express*, vol. 8, no. 4, pp. 2106–2123, Apr. 2017.

[26] Q. Péntek, S. Hein, A. Miernik, and A. Reiterer, "Image-based 3d surface approximation of the bladder using structure-from-motion for enhanced cystoscopy based on phantom data," *Biomedical Engineering / Biomedizinische Technik*, vol. 63, pp. 461 – 466, 2018.

[27] T.-B. Phan, D.-H. Trinh, D. Lamarque, D. Wolf, and C. Daul, "Dense optical flow for the reconstruction of weakly textured and structured surfaces: Application to endoscopy," in *2019 IEEE International Conference on Image Processing (ICIP)*, Sep. 2019, pp. 310–314.

[28] A. R. Widya, Y. Monno, M. Okutomi, S. Suzuki, T. Gotoda, and K. Miki, "Whole stomach 3d reconstruction and frame localization from monocular endoscope video," *IEEE Journal of Translational Engineering in Health and Medicine*, vol. 7, pp. 1–10, 2019.

[29] A. R. Widya, Y. Monno, M. Okutomi, S. Suzuki, T. Gotoda, and K. Miki, "Stomach 3d reconstruction based on virtual chromoendoscopic image generation," *2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, pp. 1848–1852, 2020.

[30] T.-B. Phan, D.-H. Trinh, D. Wolf, and C. Daul, "Optical flow-based structure-from-motion for the reconstruction of epithelial surfaces," *Pattern Recognit.*, vol. 105, no. 107391, p. 107391, Sep. 2020.

[31] C. Harris and M. Stephens, "A combined corner and edge detector," in *In Proc. of Fourth Alvey Vision Conference*, 1988, pp. 147–151.

[32] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *Proceedings of the 9th European Conference on Computer Vision - Vol-*

ume Part I, ser. ECCV'06. Berlin, Heidelberg: Springer-Verlag, 2006, pp. 430–443.

[33] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[34] R. Arandjelovic, "Three things everyone should know to improve object retrieval," in *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, ser. CVPR '12. Washington, DC, USA: IEEE Computer Society, 2012, pp. 2911–2918.

[35] A. Bursuc, G. Tolias, and H. Jégou, "Kernel local descriptors with implicit rotation matching," in *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval*, ser. ICMR '15. New York, NY, USA: ACM, 2015, pp. 595–598.

[36] J. Dong and S. Soatto, "Domain-size pooling in local descriptors: Dsp-sift," *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5097–5106, 2014.

[37] J.-W. Bian, Y.-H. Wu, J. Zhao, Y. Liu, L. Zhang, M.-M. Cheng, and I. Reid, "An evaluation of feature matchers for fundamental matrix estimation," in *British Machine Vision Conference (BMVC)*, 2019.

[38] J. L. Schonberger, H. Hardmeier, T. Sattler, and M. Pollefeys, "Comparative evaluation of hand-crafted and learned local features," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1482–1491.

[39] Y. Tian, B. Fan, and F. Wu, "L2-net: Deep learning of discriminative patch descriptor in euclidean space," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 661–669.

[40] Z. Luo, T. Shen, L. Zhou, S. Zhu, R. Zhang, Y. Yao, T. Fang, and L. Quan, "Geodesc: Learning local descriptors by integrating geometry constraints," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 168–183.

[41] A. Mishchuk, D. Mishkin, F. Radenovic, and J. Matas, "Working hard to know your neighbor's margins: Local descriptor learning loss," in *Advances in Neural Information Processing Systems*, 2017, pp. 4826–4837.

[42] E. Tola, V. Lepetit, and P. Fua, "Daisy: An efficient dense descriptor applied to wide-baseline stereo," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 5, pp. 815–830, 2009.

[43] C. B. Choy, J. Gwak, S. Savarese, and M. Chandraker, "Universal correspon-

dence network," in *Advances in Neural Information Processing Systems*, 2016, pp. 2414–2422.

[44] H. Liao, W. Lin, J. Zhang, J. Zhang, J. Luo, and S. K. Zhou, "Multiview 2d/3d rigid registration via a point-of-interest network for tracking and triangulation," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*. Computer Vision Foundation / IEEE, 2019, pp. 12 638–12 647.

[45] M. Dusmanu, I. Rocco, T. Pajdla, M. Pollefeys, J. Sivic, A. Torii, and T. Sattler, "D2-net: A trainable cnn for joint detection and description of local features," in *Proceedings of the 2019 IEEE / CVF Conference on Computer Vision and Pattern Recognition*, 2019.

[46] D. DeTone, T. Malisiewicz, and A. Rabinovich, "Superpoint: Self-supervised interest point detection and description," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 224–236.

[47] P. Truong, S. Apostolopoulos, A. Mosinska, S. Stucky, C. Ciller, and S. D. Zanet, "Glampoints: Greedily learned accurate match points," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 10 732–10 741.

[48] S. Leonard, A. Sinha, A. Reiter, M. Ishii, G. L. Gallia, R. H. Taylor *et al.*, "Evaluation and stability analysis of video-based navigation system for functional

endoscopic sinus surgery on in vivo clinical data," *IEEE_J_MI*, vol. 37, no. 10, pp. 2185–2195, Oct. 2018.

[49] O. G. Grasa, E. Bernal, S. Casado, I. Gil, and J. Montiel, "Visual slam for handheld monocular endoscope," *IEEE transactions on medical imaging*, vol. 33, no. 1, pp. 135–146, 2013.

[50] N. Mahmoud, I. Cirauqui, A. Hostettler, C. Doignon, L. Soler, J. Marescaux, and J. M. M. Montiel, "Orbslam-based endoscope tracking and 3d reconstruction," in *CARE@MICCAI*, 2016.

[51] L. Qiu and H. Ren, "Endoscope navigation and 3d reconstruction of oral cavity by visual slam with mitigated data scarcity," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, June 2018, pp. 2278–22 787.

[52] J. L. Schönberger and J.-M. Frahm, "Structure-from-motion revisited," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[53] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "Orb-slam: a versatile and accurate monocular slam system," *IEEE transactions on robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[54] C. Campos, R. Elvira, J. J. Gómez, J. M. M. Montiel, and J. D. Tardós, "ORB-

SLAM3: An accurate open-source library for visual, visual-inertial and multi-map SLAM," *arXiv preprint*, 2020.

[55] I. Khan, "Robust sparse and dense nonrigid structure from motion," *IEEE Transactions on Multimedia*, vol. 20, no. 4, pp. 841–850, April 2018.

[56] J. Lamarca, S. Parashar, A. Bartoli, and J. Montiel, "Defslam: Tracking and mapping of deforming scenes from monocular sequences," *IEEE Transactions on robotics*, vol. 37, no. 1, pp. 291–303, 2020.

[57] J. Song, J. Wang, L. Zhao, S. Huang, and G. Dissanayake, "Mis-slam: Real-time large-scale dense deformable slam system in minimal invasive surgery based on heterogeneous computing," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 4068–4075, 2018.

[58] S. H. N. Jensen, M. E. B. Doest, H. Aanæs, and A. Del Bue, "A benchmark and evaluation of non-rigid structure from motion," *International Journal of Computer Vision*, vol. 129, no. 4, pp. 882–899, 2021.

[59] C. Choy, J. Park, and V. Koltun, "Fully convolutional geometric features," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 8958–8966.

[60] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle ad-

justment - a modern synthesis," in *Proceedings of the International Workshop on Vision Algorithms: Theory and Practice*, ser. ICCV '99. Berlin, Heidelberg: Springer-Verlag, 1999, p. 298–372.

[61] Y. Lou, N. Snavely, and J. Gehrke, "MatchMiner: Efficient spanning structure mining in large image collections," in *Computer Vision – ECCV 2012*. Springer Berlin Heidelberg, 2012, pp. 45–58.

[62] M. Havlena and K. Schindler, "VocMatch: Efficient multiview correspondence for structure from motion," in *Computer Vision – ECCV 2014*. Springer International Publishing, 2014, pp. 46–60.

[63] J. L. Schönberger, A. C. Berg, and J.-M. Frahm, "PAIGE: PAirwise image geometry encoding for improved efficiency in Structure-from-Motion," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2015, pp. 1009–1018.

[64] J. Heinly, J. L. Schönberger, E. Dunn, and J.-M. Frahm, "Reconstructing the world* in six days," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2015, pp. 3287–3295.

[65] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, ISBN: 0521540518, 2004.

[66] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[67] C. Beder and R. Steffen, "Determining an initial image pair for fixing the scale of a 3D reconstruction from an image sequence," in *Lecture Notes in Computer Science*, ser. Lecture notes in computer science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 657–666.

[68] Y. Zheng, Y. Kuang, S. Sugimoto, K. Åström, and M. Okutomi, "Revisiting the PnP problem: A fast, general and optimal solution," in *2013 IEEE International Conference on Computer Vision*, Dec. 2013, pp. 2344–2351.

[69] M. Bujnak, Z. Kukelova, and T. Pajdla, "A general solution to the P4P problem for camera with unknown focal length," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, Jun. 2008.

[70] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle adjustment — a modern synthesis," in *Vision Algorithms: Theory and Practice*. Springer Berlin Heidelberg, 2000, pp. 298–372.

[71] F. Lu and R. Hartley, "A fast optimal algorithm for L2 triangulation," in *Computer Vision – ACCV 2007*. Springer Berlin Heidelberg, 2007, pp. 279–288.

[72] Li, "A practical algorithm for L triangulation with outliers," in *2007 IEEE Conference on Computer Vision and Pattern Recognition*, vol. 0, Jun. 2007, pp. 1–8.

[73] C. Aholt, S. Agarwal, and R. Thomas, "A QCQP approach to triangulation," in *Computer Vision – ECCV 2012*. Springer Berlin Heidelberg, 2012, pp. 654–667.

[74] G. P. Meyer, "An alternative probabilistic interpretation of the huber loss," *ArXiv*, vol. abs/1911.02088, 2019.

[75] J. J. Moré, "The Levenberg-Marquardt algorithm: Implementation and theory," in *Numerical Analysis*. Springer Berlin Heidelberg, 1978, pp. 105–116.

[76] S. Jégou, M. Drozdzal, D. Vazquez, A. Romero, and Y. Bengio, "The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2017, pp. 11–19.

[77] F. Wang, X. Xiang, J. Cheng, and A. L. Yuille, "Normface: l 2 hypersphere embedding for face verification," in *Proceedings of the 25th ACM international conference on Multimedia*. ACM, 2017, pp. 1041–1049.

[78] S. Honari, P. Molchanov, S. Tyree, P. Vincent, C. Pal, and J. Kautz, "Improving landmark localization with semi-supervised learning," in *Proceedings of the*

IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 1546–1555.

[79] D. R. Cox, "The regression analysis of binary sequences," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 20, no. 2, pp. 215–232, 1958.

[80] M. Menze and A. Geiger, "Object scene flow for autonomous vehicles," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[81] C. Strecha, W. Von Hansen, L. Van Gool, P. Fua, and U. Thoennessen, "On benchmarking camera calibration and multi-view stereo for high resolution imagery," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*. Ieee, 2008, pp. 1–8.

[82] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, pp. 8026–8037, 2019.

[83] H. Robbins, "A stochastic approximation method," *Annals of Mathematical Statistics*, vol. 22, pp. 400–407, 2007.

[84] L. N. Smith, "Cyclical learning rates for training neural networks," in *2017*

IEEE Winter Conference on Applications of Computer Vision (WACV). IEEE, 2017, pp. 464–472.

[85] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[86] G. A. Puerto-Souza and G. L. Mariottini, "Hierarchical multi-affine (hma) algorithm for fast and accurate feature matching in minimally-invasive surgical images," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 2007–2012.

[87] P. Moulon, P. Monasse, R. Perrot, and R. Marlet, "Openmvg: Open multiple view geometry," in *International Workshop on Reproducible Research in Pattern Recognition*. Springer, 2016, pp. 60–74.

[88] A. Baumberg, "Reliable feature matching across widely separated views," in *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No.PR00662)*, vol. 1, 2000, pp. 774–781 vol.1.

[89] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.

[90] C. Bregler, A. Hertzmann, and H. Biermann, "Recovering non-rigid 3D shape

from image streams," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 690–696, 2000.

[91] Y. Dai, H. Li, and M. He, "A simple prior-free method for non-rigid structure-from-motion factorization," *Int. J. Comput. Vis.*, vol. 107, no. 2, pp. 101–122, 2014.

[92] B. Curless and M. Levoy, "A volumetric method for building complex models from range images," in *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques - SIGGRAPH '96*. New York, New York, USA: ACM Press, 1996.

[93] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab, "Deeper depth prediction with fully convolutional residual networks," in *2016 4th Int. Conf. 3D Vis.*, Oct. 2016, pp. 239–248.

[94] M. Visentini-Scarzanella, T. Sugiura, T. Kaneko, and S. Koto, "Deep monocular 3d reconstruction for assisted navigation in bronchoscopy," *International journal of computer assisted radiology and surgery*, vol. 12, no. 7, pp. 1089–1099, 2017.

[95] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," 2014.

[96] F. Mahmood and N. J. Durr, "Deep learning and conditional random fields-based depth estimation and topographical reconstruction from conventional endoscopy," *Medical image analysis*, vol. 48, pp. 230–243, 2018.

[97] S.-P. Yang, J.-J. Kim, K.-W. Jang, W.-K. Song, and K.-H. Jeong, "Compact stereo endoscopic camera using microprism arrays," *Optics letters*, vol. 41, no. 6, pp. 1285–1288, 2016.

[98] M. Simi, M. Silvestri, C. Cavallotti, M. Vatteroni, P. Valdastri, A. Menciassi *et al.*, "Magnetically activated stereoscopic vision system for laparoendoscopic single-site surgery," *IEEE_J_MECH*, vol. 18, no. 3, pp. 1140–1151, 2013.

[99] R. Garg, V. K. BG, G. Carneiro, and I. Reid, "Unsupervised cnn for single view depth estimation: Geometry to the rescue," in *Comput. Vis. ECCV*. Springer, 2016, pp. 740–756.

[100] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, "Unsupervised learning of depth and ego-motion from video," in *2017 IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 6612–6619.

[101] Z. Yin and J. Shi, "Geonet: Unsupervised learning of dense depth, optical flow and camera pose," in *2018 IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 1983–1992.

[102] R. Mahjourian, M. Wicke, and A. Angelova, "Unsupervised learning of depth and ego-motion from monocular video using 3d geometric constraints," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 5667–5675.

[103] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," in *Adv. Neural Inf. Process. Syst. 27*. Curran Associates, Inc., 2014, pp. 2366–2374.

[104] M. Kazhdan, M. Bolitho, and H. Hoppe, "Poisson surface reconstruction," in *Proceedings of the fourth Eurographics symposium on Geometry processing*, vol. 7, 2006.

[105] M. Turan, Y. Y. Pilavci, I. Ganiyusufoglu, H. Araujo, E. Konukoglu, and M. Sitti, "Sparse-then-dense alignment-based 3d map reconstruction method for endoscopic capsule robots," *Machine Vision and Applications*, vol. 29, no. 2, pp. 345–359, 2018.

[106] H. N. Tokgozoglu, E. M. Meisner, M. Kazhdan, and G. D. Hager, "Color-based hybrid reconstruction for endoscopy," in *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, June 2012, pp. 8–15.

[107] A. Karargyris and N. Bourbakis, "Three-dimensional reconstruction of the di-

gestive wall in capsule endoscopy videos using elastic video interpolation," *IEEE Transactions on Medical Imaging*, vol. 30, no. 4, pp. 957–971, April 2011.

[108] Q. Zhao, T. Price, S. Pizer, M. Niethammer, R. Alterovitz, and J. Rosenman, "The endoscopogram: A 3d model reconstructed from endoscopic video frames," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016*, S. Ourselin, L. Joskowicz, M. R. Sabuncu, G. Unal, and W. Wells, Eds. Cham: Springer International Publishing, 2016, pp. 439–447.

[109] N. Mahmoud, A. Hostettler, T. Collins, L. Soler, C. Doignon, and J. Montiel, "Slam based quasi dense reconstruction for minimally invasive surgery scenes," *arXiv preprint*, 2017.

[110] R. Ma, R. Wang, S. Pizer, J. Rosenman, S. K. McGill, and J.-M. Frahm, "Real-time 3d reconstruction of colonoscopic surfaces for determining missing regions," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2019, pp. 573–582.

[111] R. J. Chen, T. L. Bobrow, T. Athey, F. Mahmood, and N. J. Durr, "Slam endoscopy enhanced by adversarial depth prediction," *arXiv preprint*, 2019.

[112] X. Liu, A. Sinha, M. Unberath, M. Ishii, G. Hager, R. Taylor *et al.*, "Self-supervised learning for dense depth estimation in monocular endoscopy," in *OR*

2.0 Context Aware Oper. Theaters Comput. Assist. Robot. Endosc. Clin. Image Based Proced. Skin Image Anal. Springer Verlag, 2018, pp. 128–138.

[113] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *Proc. 2005 IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 1. Washington, DC, USA: IEEE Computer Society, 2005, pp. 539–546.

[114] A. Odena, V. Dumoulin, and C. Olah, "Deconvolution and checkerboard artifacts," *Distill*, vol. 1, no. 10, p. e3, 2016.

[115] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu, "Spatial transformer networks," in *Proc. 28th Int. Conf. Neural Inf. Process. Syst.*, vol. 2. Cambridge, MA, USA: MIT Press, 2015, pp. 2017–2025.

[116] E. Hüllermeier and W. Waegeman, "Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods," *Mach. Learn.*, vol. 110, no. 3, pp. 457–506, Mar. 2021.

[117] D. R. Canelhas, "Truncated signed distance fields applied to robotics," Ph.D. dissertation, School of Science and Technology, 2017.

[118] C. Zach, T. Pock, and H. Bischof, "A globally optimal algorithm for robust tv-l 1 range image integration," in *ICCV*. IEEE, 2007, pp. 1–8.

[119] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3d surface construction algorithm," *ACM siggraph computer graphics*, vol. 21, no. 4, pp. 163–169, 1987.

[120] S. Billings and R. Taylor, "Generalized iterative most likely oriented-point (g-imlop) registration," *IJCARS*, vol. 10, no. 8, pp. 1213–1226, 2015.

[121] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[122] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, and G. Taubin, "The ball-pivoting algorithm for surface reconstruction," *IEEE Transactions on Visualization and Computer Graphics*, vol. 5, no. 4, pp. 349–359, Oct 1999.

[123] F. Cazals and J. Giesen, "Delaunay triangulation based surface reconstruction," in *Effective computational geometry for curves and surfaces*. Springer, 2006, pp. 231–276.

[124] R. Wang, S. M. Pizer, and J.-M. Frahm, "Recurrent neural network for (un-) supervised learning of monocular video visual odometry and depth," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5555–5564.

[125] X. Huang, G. Mei, J. Zhang, and R. Abbas, "A comprehensive survey on point cloud registration," *CoRR*, vol. abs/2103.02690, 2021.

[126] A. Kovnatsky, M. M. Bronstein, A. M. Bronstein, K. Glashoff, and R. Kimmel, "Coupled quasi-harmonic bases," in *Computer Graphics Forum*, vol. 32, no. 2pt4. Wiley Online Library, 2013, pp. 439–448.

[127] Q. Huang, F. Wang, and L. Guibas, "Functional map networks for analyzing and exploring large shape collections," *ACM Transactions on Graphics (TOG)*, vol. 33, no. 4, pp. 1–11, 2014.

[128] Y. Aflalo, A. Dubrovina, and R. Kimmel, "Spectral generalized multi-dimensional scaling," *International Journal of Computer Vision*, vol. 118, no. 3, pp. 380–392, 2016.

[129] D. Eynard, E. Rodola, K. Glashoff, and M. M. Bronstein, "Coupled functional maps," in *2016 Fourth International Conference on 3D Vision (3DV)*. IEEE, 2016, pp. 399–407.

[130] O. Burghard, A. Dieckmann, and R. Klein, "Embedding shapes with green's functions for global shape matching," *Computers & Graphics*, vol. 68, pp. 1–10, 2017.

[131] E. Rodolà, L. Cosmo, M. M. Bronstein, A. Torsello, and D. Cremers, "Partial

functional correspondence," in *Computer Graphics Forum*, vol. 36, no. 1. Wiley Online Library, 2017, pp. 222–236.

[132] R. Litman and A. M. Bronstein, "Learning spectral descriptors for deformable shape correspondence," *IEEE transactions on pattern analysis and machine intelligence*, vol. 36, no. 1, pp. 171–180, 2013.

[133] T. Windheuser, M. Vestner, E. Rodola, R. Triebel, and D. Cremers, "Optimal intrinsic descriptors for non-rigid shape analysis," in *Proceedings of the British Machine Vision Conference*. BMVA Press, 2014.

[134] D. Boscaini, J. Masci, S. Melzi, M. M. Bronstein, U. Castellani, and P. Vandergheynst, "Learning class-specific descriptors for deformable shapes using localized spectral convolutional networks," in *Computer Graphics Forum*, vol. 34, no. 5. Wiley Online Library, 2015, pp. 13–23.

[135] D. Boscaini, J. Masci, E. Rodolà, M. M. Bronstein, and D. Cremers, "Anisotropic diffusion descriptors," in *Computer Graphics Forum*, vol. 35, no. 2. Wiley Online Library, 2016, pp. 431–441.

[136] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, and T. Funkhouser, "3dmatch: Learning local geometric descriptors from rgb-d reconstructions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1802–1811.

[137] M. Khoury, Q.-Y. Zhou, and V. Koltun, "Learning compact geometric features," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 153–161.

[138] H. Deng, T. Birdal, and S. Ilic, "Ppfnet: Global context aware local features for robust 3d point matching," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 195–205.

[139] ——, "Ppf-foldnet: Unsupervised learning of rotation invariant 3d local descriptors," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 602–618.

[140] Z. Gojcic, C. Zhou, J. D. Wegner, and A. Wieser, "The perfect match: 3d point cloud matching with smoothed densities," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5545–5554.

[141] X. Bai, Z. Luo, L. Zhou, H. Fu, L. Quan, and C.-L. Tai, "D3feat: Joint learning of dense detection and description of 3d local features," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[142] D. Boscaini, J. Masci, E. Rodolà, and M. Bronstein, "Learning shape correspondence with anisotropic convolutional neural networks," in *Advances in neural information processing systems*, 2016, pp. 3189–3197.

[143] O. Litany, T. Remez, E. Rodola, A. Bronstein, and M. Bronstein, "Deep functional maps: Structured prediction for dense shape correspondence," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 5659–5667.

[144] F. Monti, D. Boscaini, J. Masci, E. Rodola, J. Svoboda, and M. M. Bronstein, "Geometric deep learning on graphs and manifolds using mixture model cnns," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5115–5124.

[145] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: going beyond euclidean data," *IEEE Signal Processing Magazine*, vol. 34, no. 4, pp. 18–42, 2017.

[146] N. Donati, A. Sharma, and M. Ovsjanikov, "Deep geometric functional maps: Robust feature learning for shape correspondence," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 8592–8601.

[147] T. Groueix, M. Fisher, V. G. Kim, B. C. Russell, and M. Aubry, "3d-coded: 3d correspondences by deep deformation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 230–246.

[148] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the

kitti vision benchmark suite," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 3354–3361.

[149] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *CoRR*, vol. abs/1502.03167, 2015.

[150] S. Santurkar, D. Tsipras, A. Ilyas, and A. Madry, "How does batch normalization help optimization?" in *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., vol. 31. Curran Associates, Inc., 2018, pp. 2483–2493.

[151] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," 2016.

[152] D. Ulyanov, A. Vedaldi, and V. S. Lempitsky, "Instance normalization: The missing ingredient for fast stylization," *CoRR*, vol. abs/1607.08022, 2016.

[153] Y. Wu and K. He, "Group normalization," in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[154] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[155] A. Ortiz, C. Robinson, D. Morris, O. Fuentes, C. Kiekintveld, M. M. Hassan, and N. Jojic, "Local context normalization: Revisiting local normalization," in *Pro-*

ceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2020.

[156] H. Nam and H.-E. Kim, "Batch-instance normalization for adaptively style-invariant neural networks," in *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., vol. 31. Curran Associates, Inc., 2018, pp. 2558–2567.

[157] P. Luo, J. Ren, Z. Peng, R. Zhang, and J. Li, "Differentiable learning-to-normalize via switchable normalization," in *International Conference on Learning Representations*, 2019.

[158] W. Shao, T. Meng, J. Li, R. Zhang, Y. Li, X. Wang, and P. Luo, "Ssn: Learning sparse switchable normalization via sparsestmax," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[159] D. Maturana and S. Scherer, "Voxnet: A 3d convolutional neural network for real-time object recognition," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 922–928.

[160] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1912–1920.

[161] X. Roynard, J.-E. Deschaud, and F. Goulette, "Classification of Point Cloud for Road Scene Understanding with Multiscale Voxel Deep Network," in *10th workshop on Planning, Perceptionand Navigation for Intelligent Vehicules PPNIV'2018*, Madrid, Spain, Oct. 2018, preprint.

[162] B. Graham, M. Engelcke, and L. Van Der Maaten, "3d semantic segmentation with submanifold sparse convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 9224–9232.

[163] C. Choy, J. Gwak, and S. Savarese, "4d spatio-temporal convnets: Minkowski convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3075–3084.

[164] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[165] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017, pp. 5099–5108.

[166] M. Atzmon, H. Maron, and Y. Lipman, "Point convolutional neural networks by

extension operators," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 4, Jul. 2018.

[167] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "Pointcnn: Convolution on x-transformed points," in *Advances in neural information processing systems*, 2018, pp. 820–830.

[168] P. Hermosilla, T. Ritschel, P.-P. Vázquez, À. Vinacua, and T. Ropinski, "Monte carlo convolution for learning on non-uniformly sampled point clouds," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 6, pp. 1–12, 2018.

[169] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. J. Guibas, "Kpconv: Flexible and deformable convolution for point clouds," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[170] J. Masci, D. Boscaini, M. M. Bronstein, and P. Vandergheynst, "Geodesic convolutional neural networks on riemannian manifolds," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops*, December 2015.

[171] J. Huang, H. Zhang, L. Yi, T. Funkhouser, M. Niessner, and L. J. Guibas, "Texturenet: Consistent local parametrizations for learning from high-resolution

signals on meshes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[172] R. Hanocka, A. Hertz, N. Fish, R. Giryes, S. Fleishman, and D. Cohen-Or, "Meshcnn: a network with an edge," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 4, pp. 1–12, 2019.

[173] L. Landrieu and M. Simonovsky, "Large-scale point cloud semantic segmentation with superpoint graphs," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[174] L. Jiang, H. Zhao, S. Liu, X. Shen, C.-W. Fu, and J. Jia, "Hierarchical point-edge interaction network for point cloud semantic segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[175] G. Li, M. Muller, A. Thabet, and B. Ghanem, "Deepgcns: Can gcns go as deep as cnns?" in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[176] J. Schult, F. Engelmann, T. Kontogianni, and B. Leibe, "Dualconvmesh-net: Joint geodesic and euclidean convolutions on 3d meshes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[177] H. Lei, N. Akhtar, and A. Mian, "Spherical kernel for efficient graph convolution on 3d point clouds," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2020.

[178] P. Besl and N. D. McKay, "A method for registration of 3-d shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992.

[179] S. Rusinkiewicz and M. Levoy, "Efficient variants of the ICP algorithm," in *Proceedings Third International Conference on 3-D Digital Imaging and Modeling*, May 2001, pp. 145–152.

[180] A. W. Fitzgibbon, "Robust registration of 2d and 3d point sets," *Image and Vision Computing*, vol. 21, no. 13, pp. 1145–1153, 2003, british Machine Vision Computing 2001.

[181] S. D. Billings, E. M. Boctor, and R. H. Taylor, "Iterative Most-Likely Point Registration (IMLP): A Robust Algorithm for Computing Optimal Shape Alignment," *PLoS ONE*, vol. 10, no. 3, p. e0117688, Mar. 2015.

[182] A. Segal, D. Haehnel, and S. Thrun, "Generalized-icp." in *Robotics: science and systems*, vol. 2, no. 4. Seattle, WA, 2009, p. 435.

[183] O. Duchenne, F. Bach, I.-S. Kweon, and J. Ponce, "A tensor-based algorithm for

high-order graph matching," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 12, pp. 2383–2395, 2011.

[184] F. Zhou and F. De la Torre, "Factorized graph matching," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 127–134.

[185] L. Livi and A. Rizzi, "The graph matching problem," *Pattern Analysis and Applications*, vol. 16, no. 3, pp. 253–283, 2013.

[186] F. Zhou and F. De la Torre, "Factorized graph matching," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 9, pp. 1774–1789, 2016.

[187] A. Myronenko and X. Song, "Point set registration: Coherent point drift," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 12, pp. 2262–2275, 2010.

[188] A. Rasoulian, R. Rohling, and P. Abolmaesumi, "Group-wise registration of point sets for statistical shape models," *IEEE Transactions on Medical Imaging*, vol. 31, no. 11, pp. 2025–2034, 2012.

[189] J. Fan, J. Yang, D. Ai, L. Xia, Y. Zhao, X. Gao, and Y. Wang, "Convex hull indexed gaussian mixture model (ch-gmm) for 3d point set registration," *Pattern Recognition*, vol. 59, pp. 126–141, 2016, compositional Models and Structured Learning for Visual Recognition.

[190] G. D. Evangelidis, D. Kounades-Bastian, R. Horaud, and E. Z. Psarakis, "A Generative Model for the Joint Registration of Multiple Point Sets," in *Computer Vision – ECCV 2014*, ser. Lecture Notes in Computer Science, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014, pp. 109–122.

[191] O. Enqvist, K. Josephson, and F. Kahl, "Optimal correspondences from pairwise constraints," in *2009 IEEE 12th international conference on computer vision*. IEEE, 2009, pp. 1295–1302.

[192] N. Dym, H. Maron, and Y. Lipman, "Ds++: A flexible, scalable and provably tight relaxation for matching problems," *arXiv preprint*, 2017.

[193] X. Huang, J. Zhang, Q. Wu, L. Fan, and C. Yuan, "A coarse-to-fine algorithm for matching and registration in 3d cross-source point clouds," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 10, pp. 2965–2977, 2017.

[194] H. M. Le, T.-T. Do, T. Hoang, and N.-M. Cheung, "Sdrsac: Semidefinite-based randomized approach for robust point cloud registration without correspondences," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 124–133.

[195] J. P. Iglesias, C. Olsson, and F. Kahl, "Global optimality for point set registra-

tion using semidefinite programming," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[196] H. Yang, J. Shi, and L. Carlone, "Teaser: Fast and certifiable point cloud registration," *IEEE Transactions on Robotics*, vol. 37, no. 2, pp. 314–333, 2020.

[197] J. Li, Q. Hu, and M. Ai, "Point cloud registration based on one-point ransac and scale-annealing biweight estimation," *IEEE Transactions on Geoscience and Remote Sensing*, pp. 1–14, 2021.

[198] H. Maron, N. Dym, I. Kezurer, S. Kovalsky, and Y. Lipman, "Point registration via efficient convex relaxation," *ACM Transactions on Graphics (TOG)*, vol. 35, no. 4, pp. 1–12, 2016.

[199] H. Deng, T. Birdal, and S. Ilic, "3D local features for direct pairwise registration," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, Jun. 2019.

[200] W. Lu, G. Wan, Y. Zhou, X. Fu, P. Yuan, and S. Song, "Deepvcp: An end-to-end deep neural network for point cloud registration," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 12–21.

[201] G. D. Pais, S. Ramalingam, V. M. Govindu, J. C. Nascimento, R. Chellappa, and P. Miraldo, "3dregnet: A deep neural network for 3d point registration,"

in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 7193–7203.

[202] G. Riegler, A. Osman Ulusoy, and A. Geiger, "Octnet: Learning deep 3d representations at high resolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 3577–3586.

[203] M. Tatarchenko, A. Dosovitskiy, and T. Brox, "Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2088–2096.

[204] P.-S. Wang, Y. Liu, Y.-X. Guo, C.-Y. Sun, and X. Tong, "O-cnn: Octree-based convolutional neural networks for 3d shape analysis," *ACM Transactions On Graphics (TOG)*, vol. 36, no. 4, pp. 1–11, 2017.

[205] Z. J. Yew and G. H. Lee, "3dfeat-net: Weakly supervised local 3d features for point cloud registration," in *Computer Vision – ECCV 2018*, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds. Cham: Springer International Publishing, 2018, pp. 630–646.

[206] Y. Wang and J. M. Solomon, "Deep closest point: Learning representations for point cloud registration," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 3523–3532.

[207] J. Zhou, M. Wang, W. Mao, M. Gong, and X. Liu, "Siamesepointnet: A siamese point network architecture for learning 3d shape descriptor," in *Computer Graphics Forum*, vol. 39, no. 1. Wiley Online Library, 2020, pp. 309–321.

[208] Z. J. Yew and G. H. Lee, "Rpm-net: Robust point matching using learned features," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 824–11 833.

[209] C. Choy, W. Dong, and V. Koltun, "Deep global registration," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 2514–2523.

[210] W. Yuan, B. Eckart, K. Kim, V. Jampani, D. Fox, and J. Kautz, "Deepgmr: Learning latent gaussian mixture models for registration," in *European Conference on Computer Vision*. Springer, 2020, pp. 733–750.

[211] Y. Aoki, H. Goforth, R. A. Srivatsan, and S. Lucey, "Pointnetlk: Robust & efficient point cloud registration using pointnet," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7163–7172.

[212] X. Huang, G. Mei, and J. Zhang, "Feature-metric registration: A fast semi-supervised approach for robust point cloud registration without correspondences," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 366–11 374.

[213] H. Zhu, C. Cui, L. Deng, R. C. Cheung, and H. Yan, "Elastic net constraint-based tensor model for high-order graph matching," *IEEE transactions on cybernetics*, 2019.

[214] S. D. Billings, A. Sinha, A. Reiter, S. Leonard, M. Ishii, G. D. Hager, and R. H. Taylor, "Anatomically constrained video-ct registration via the v-imlop algorithm," in *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2016*, S. Ourselin, L. Joskowicz, M. R. Sabuncu, G. Unal, and W. Wells, Eds. Cham: Springer International Publishing, 2016, pp. 133–141.

[215] K. Clark, B. Vendt, K. Smith, J. Freymann, J. Kirby, P. Koppel, S. Moore, S. Phillips, D. Maffitt, M. Pringle *et al.*, "The cancer imaging archive (tcia): maintaining and operating a public information repository," *Journal of digital imaging*, vol. 26, no. 6, pp. 1045–1057, 2013.

[216] A. Sinha, A. Reiter, S. Leonard, M. Ishii, G. D. Hager, and R. H. Taylor, "Simultaneous segmentation and correspondence improvement using statistical modes," in *Medical Imaging 2017: Image Processing*, M. A. Styner and E. D. Angelini, Eds., vol. 10133, International Society for Optics and Photonics. SPIE, 2017, pp. 377 – 384.

[217] S. Valette, J. M. Chassery, and R. Prost, "Generic remeshing of 3d triangular

meshes with metric-dependent discrete voronoi diagrams," *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 2, pp. 369–381, 2008.

[218] Z. Gojcic, C. Zhou, J. D. Wegner, and A. Wieser, "The perfect match: 3d point cloud matching with smoothed densities," in *Proceedings of the IEEE / CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5545–5554.

[219] S. Ehsan, A. F. Clark, N. U. Rehman, and K. D. McDonald-Maier, "Integral images: Efficient algorithms for their computation and storage in resource-constrained embedded vision systems," *Sensors*, vol. 15, no. 7, pp. 16 804–16 830, 2015.

[220] Z. Liu, H. Hu, Y. Cao, Z. Zhang, and X. Tong, "A closer look at local aggregation operators in point cloud analysis," *ECCV*, 2020.

[221] L. McInnes, J. Healy, N. Saul, and L. Großberger, "Umap: Uniform manifold approximation and projection," *Journal of Open Source Software*, vol. 3, no. 29, p. 861, 2018.

[222] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-Scale direct monocular SLAM," *Lect. Notes Comput. Sci.*, vol. 8690 LNCS, no. PART 2, pp. 834–849, 2014.

[223] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid,

and J. J. Leonard, “Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age,” *IEEE Trans. Rob.*, vol. 32, no. 6, pp. 1309–1332, 2016.

[224] K. Tateno, F. Tombari, I. Laina, and N. Navab, “CNN-SLAM: Real-time dense monocular SLAM with learned depth prediction,” *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 6565–6574, 2017.

[225] R. Mur-Artal and J. D. Tardós, “ORB-SLAM2 an Open-Source SLAM system for monocular stereo.pdf,” *IEEE Trans. Rob.*, vol. 33, no. 5, pp. 1255–1262, 2017.

[226] R. Li, S. Wang, and D. Gu, “Ongoing evolution of visual SLAM from geometry to deep learning: Challenges and opportunities,” *Cognit. Comput.*, vol. 10, no. 6, pp. 875–889, Dec. 2018.

[227] M. Bloesch, J. Czarnowski, R. Clark, S. Leutenegger, and A. J. Davison, “CodeSLAM - learning a compact, optimisable representation for dense visual SLAM,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2560–2568, 2018.

[228] T. Laidlow, J. Czarnowski, and S. Leutenegger, “DeepFusion: Real-time dense 3D reconstruction for monocular SLAM using single-view depth and gradient

predictions," *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2019-May, pp. 4068–4074, 2019.

[229] L. Tiwari, P. Ji, Q.-H. Tran, B. Zhuang, S. Anand, and M. Chandraker, "Pseudo rgb-d for self-improving monocular slam and depth prediction," in *European Conference on Computer Vision*, 2020.

[230] W. N. Greene and N. Roy, "Metrically-Scaled monocular SLAM using learned scale factors," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2020, pp. 43–50.

[231] J. Czarnowski, T. Laidlow, R. Clark, and A. J. Davison, "DeepFactors: Real-Time probabilistic dense monocular SLAM," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 721–728, 2020.

[232] C. Wang, M. Oda, Y. Hayashi, T. Kitasaka, H. Honma, H. Takabatake, M. Mori, H. Natori, and K. Mori, "Visual slam for bronchoscope tracking and bronchus reconstruction in bronchoscopic navigation," in *Medical Imaging 2019: Image-Guided Procedures, Robotic Interventions, and Modeling*, vol. 10951. International Society for Optics and Photonics, 2019, p. 109510A.

[233] C. Xie, T. Yao, J. Wang, and Q. Liu, "Endoscope localization and gastrointestinal feature map construction based on monocular SLAM technology," *J. Infect. Public Health*, pp. 4–11, 2019.

[234] R. Ma, R. Wang, Y. Zhang, S. Pizer, S. K. McGill, J. Rosenman, and J.-M. Frahm, "RNNSLAM: Reconstructing the 3D colon to visualize missing regions during a colonoscopy," *Med. Image Anal.*, vol. 72, p. 102100, May 2021.

[235] N. Yang, L. v. Stumberg, R. Wang, and D. Cremers, "D3vo: Deep depth, deep pose and deep uncertainty for monocular visual odometry," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1281–1292.

[236] C. Tang and P. Tan, "Ba-net: Dense bundle adjustment network," *arXiv preprint*, 2018.

[237] X. Wei, Y. Zhang, Z. Li, Y. Fu, and X. Xue, "Deepsfm: Structure from motion via deep bundle adjustment," in *European conference on computer vision*. Springer, 2020, pp. 230–247.

[238] H. Zhan, C. S. Weerasekera, J. W. Bian, and I. Reid, "Visual odometry revisited: What should be learnt?" *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 4203–4210, 2020.

[239] N. Mahmoud, I. Cirauqui, A. Hostettler, C. Doignon, L. Soler, J. Marescaux, and J. M. M. Montiel, "Orbslam-based endoscope tracking and 3d reconstruction," in *Computer-Assisted and Robotic Endoscopy*, T. Peters, G.-Z. Yang, N. Navab,

K. Mori, X. Luo, T. Reichl, and J. McLeod, Eds. Cham: Springer International Publishing, 2017, pp. 72–83.

[240] N. Mahmoud, T. Collins, A. Hostettler, L. Soler, C. Doignon, and J. Montiel, "Live tracking and dense reconstruction for handheld monocular endoscopy," *IEEE Transactions on Medical Imaging*, vol. 38, pp. 79–89, 2019.

[241] M. Turan, E. P. Örnek, N. Ibrahimli, C. Giracoglu, Y. Almalioglu, M. Yanik, and M. Sitti, "Unsupervised odometry and depth learning for endoscopic capsule robots," *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1801–1807, 2018.

[242] M. Turan, Y. Almalioglu, H. Araujo, E. Konukoglu, and M. Sitti, "A non-rigid map fusion-based direct SLAM method for endoscopic capsule robots," *Int J Intell Robot Appl*, vol. 1, no. 4, pp. 399–409, Nov. 2017.

[243] J. Song, L. Zhao, S. Huang, and G. Dissanayake, "An observable time series based slam algorithm for deforming environment," *ArXiv*, vol. abs/1906.08563, 2019.

[244] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," *Lect. Notes Comput. Sci.*, vol. 9351, pp. 234–241, 2015.

[245] G. Liu, F. A. Reda, K. J. Shih, T.-C. Wang, A. Tao, and B. Catanzaro, "Image inpainting for irregular holes using partial convolutions," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 85–100.

[246] M. Bosse, G. Agamennoni, I. Gilitschenski *et al.*, *Robust estimation and applications in robotics*. Now Publishers, 2016.

[247] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley, "Least squares generative adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2794–2802.

[248] M. Avi-Aharon, A. Arbelle, and T. R. Raviv, "Deephist: Differentiable joint and color histogram layers for image-to-image translation," 2020.

[249] J. Blanco, "A tutorial on se (3) transformation parameterizations and on-manifold optimization," *University of Malaga, Tech. Rep*, no. 3, 2010.

[250] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "ISAM2: Incremental smoothing and mapping using the bayes tree," *Int. J. Rob. Res.*, vol. 31, no. 2, pp. 216–235, 2012.

[251] D. Gálvez-López and J. D. Tardós, "Bags of binary words for fast place recognition in image sequences," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188–1197, October 2012.

[252] Z. Zhang and D. Scaramuzza, "A tutorial on quantitative trajectory evaluation for visual(-inertial) odometry," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 7244–7251.

[253] J. Kopf, X. Rong, and J.-B. Huang, "Robust consistent video depth estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 1611–1621.